İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

UYARLAMALI SÜZGEÇLER İÇİN YENİ BİR STOKASTİK KONTROL ALGORİTMASI VE SİSTEM TANILAMA UYGULAMASI

DOKTORA TEZİ

Y. Müh. Osman Hilmi Koçal

Tezin Enstitüye Verildiği Tarih : 20 Eylül 1996
Tezin Savunulduğu Tarih : 06 Mart 1997

Ana Bilim Dalı : ELEKTRONİK VE HABERLEŞME

Programı : ELEKTRONİK VE HABERLEŞME

Tez Danışmanı : Prof. Dr. Enise Erimez 1.4.97
Diğer Jüri Üyeleri : Prof. Dr. Ergül Akçakaya 1.4.97
Prof. Dr. Leyla Gören 2.4.97
Prof. Dr. Avni Morgül 3.4.97
Prof. Dr. Atilla Ataman 2.4.97

MART 1997

ÖNSÖZ

Bu tez çalışması sırasında teorik bilgisi , sezgisel çıkarım ve yönlendirmeleriyle yardımcı olan ve teşvik eden sayın hocam Prof. Hasan Önal , Prof. Dr. Enise Erimez ve Yrd. Doç. Dr. Külmiz Çevik'e , paket programların kullanılmasındaki yardımlarından dolayı Kontrol-Kumanda Ana Bilim Dalı araştırma görevlilerine , maddi ve manevi destek sağlayan aileme teşekkür ederim.

Mart 1997

Osman Hilmi Koçal

İÇİNDEKİLER

NOTASYON LİSTESİ

n	:	Ayrık zaman değişkeni
M	:	Uyarlamalı süzgeç derecesi
$x(n)$	:	Uyarlamalı süzgeç giriş işareti
$\mathbf{x}(n)$	:	Giriş işaret vektörü
$y(n)$	:	Uyarlamalı süzgeç çıkış işareti
$d(n)$	:	Öngörülmesi istenen işaret
σ_d^2	:	İstenen işaretin varyansı
$e(n)$	:	Çıkış işareti ile istenen işaret farkından oluşan hata işareti
$w_k(n)$	:	k. süzgeç katsayısı
$\mathbf{w}(n)$	:	Süzgeç katsayı vektörü
$\mathbf{w}_0$	:	Optimum katsayı vektörü
$\mathbf{e}_w(n)$	:	Katsayı hata vektörü
$f(.)$	:	Olasılık yoğunluk fonksiyonu
$E[.]$	:	Ortalama değer işlemcisi
∇_w	:	Katsayı vektörüne göre gradient işlemcisi
$\nabla_w(n)$	:	Katsayı vektörüne göre gradient işlemcisinin n anındaki değeri
μ	:	Adım parametresi
$\mathbf{R}$	:	Giriş işaretinin özilişki matrisi
$\mathbf{p}$	:	Giriş işareti ile istenen işaret arasındaki çapraz ilişki vektörü
Λ	:	Özilişki matrisinin özdeğerlerinden oluşan diagonal matris

$\mathbf{Q}$	:	Sütunları öilişki matrisinin özvektörleri olan matris
λ_I	:	Özilişki matrisinin i özdeğeri
$\mathbf{R}_{ort}$	:	Özilişki matrisi için örnek ortalaması kullanılarak bulunan öngörü matrisi
$\mathbf{R}_{ort}(n)$	:	Örnek boyutu n olan bir küme için örnek ortalaması matrisi
$\mathbf{R}(n)$	:	Örnek boyutu n olan bir küme için öngörü matrisi
$\mathbf{R}_{ani}(n)$	:	Özilişki matrisi için n anındaki değerler kullanılarak bulunan öngörü matrisi
$\mathbf{p}_{ort}$	:	Çapraz ilişki vektörü için örnek ortalaması kullanılarak bulunan öngörü vektörü
$\mathbf{p}_{ort}(n)$	:	Örnek boyutu n olan bir küme için örnek ortalaması vektörü
$\mathbf{p}(n)$	:	Örnek boyutu n olan bir küme için öngörü vektörü
$\mathbf{p}_{ani}(n)$	:	Çapraz ilişki vektörü için n anındaki değerler kullanılarak bulunan öngörü vektörü
$\mathbf{R}_D$	:	Köşegen özilişki matrisi
$\mathbf{R}_D(n)$	:	Örnek boyutu n olan bir küme için köşegen özilişki öngörü matrisi
$\mathbf{R}_L$	:	Altüçgen özilişki matrisi
$\mathbf{R}_L(n)$	:	Örnek boyutu n olan bir küme için altüçgen özilişki öngörü matrisi
$\mathbf{R}_U$	:	Üstüçgen özilişki matrisi
$\mathbf{R}_U(n)$	:	Örnek boyutu n olan bir küme için üstüçgen özilişki öngörü matrisi
r_i	:	i. özilişki katsayısı
$r_i(n)$	:	Örnek boyutu n olan bir küme için i. özilişki katsayısı öngörü değeri
p_i	:	i. çapraz ilişki katsayısı

$p_i(n)$	:	Örnek boyutu n olan bir küme için i. çapraz ilişki katsayısı öngörü değeri
k(n)	:	Katsayı hata vektörünün n örnek üzerinden hesaplanmış ortalamasının normu
LMS	:	En küçük karesel ortalama algoritması
RLS	:	Ardışıl en küçük kareler algoritması
UGS	:	Uyarlamalı Gauss-Seidel algoritması
madpr	:	Bir algoritma adımındaki çarpma ve bölme işlemi sayısı
T	:	Örnekleme peryodu

ŞEKİL LİSTESİ

TABLO LİSTESİ

ÖZET

Uyarlamalı FIR süzgeçlerde kullanılan kontrol algoritmaları , gradient tabanlı algoritmalar ve ardışıl algoritmalar olmak üzere iki ana sınıfta toplanabilir. Gradient tabanlı algoritmalarda ilgilenilen amaç ölçütünün gradient vektörü kullanılır ve yakınsama hızı özilişki matrisinin yapısına bağımlıdır. Çok kullanılan bir gradient tabanlı algoritma LMS (least-mean-square) algoritmasıdır. Ardışıl algoritmalarda özilişki matrisinin tersi her adımda ardışıl olarak hesaplanır ve yakınsama hızı özilişki matrisinin yapısından bağımsızdır. Bunun sonucu olarak yakınsama hızı gradient tabanlı algoritmalara göre çok daha büyüktür. İyi bilinen bir ardışıl algoritma RLS (recursive least square) algoritmasıdır. Algoritmaların karşılaştırılmasında önemli ölçütlerden biri de madpr (multiplication and division per recusion) olarak adlandırılan bir iterasyon adımındaki çarpma ve bölme sayısıdır. Süzgeç katsayı vektörünün boyutu M olmak üzere , LMS algoritmasında madpr M ile doğrusal artarken , RLS algoritmasında karesel olarak artmaktadır.

Bu çalışmada uyarlamalı FIR süzgeçler için tabanını doğrusal denklem sistemlerinin çözümünde kullanılan ardışıl yöntemlerin oluşturduğu yeni bir stokastik kontrol algoritması önerilmiştir. Optimum süzgeç katsayılarının elde edildiği Wiener-Hopf denklemlerine Jacobi ve Gauss-Seidel algoritmaları ilk kez uygulanmış ve süzgeç katsayılarını ardışıl olarak hesaplayan iki yöntem bulunmuştur. Jacobi algoritması kullanılarak bulunan yöntemin kararlı olması için gradient tabanlı algoritmalarda kararlılık bakımından sağlanması gereken koşula benzer bir koşulun sağlanması gerektiği görülmüştür. Gauss-Seidel algoritması içeren yöntemin bir koşul gerektirmeden yakınsayacağı anlaşılmıştır. Bu yöntemde özilişki matrisi ve çapraz ilişki vektörü yerine bunların öngörü değerleri olan örnek ortalamaları kullanılarak yeni bir stokastik kontrol algoritması elde edilmiştir. Stokastik kontrol algoritması kullanılarak hesaplanan süzgeç katsayılarıyla ilgilenilen sürecin özilişki katsayılarının istatistiksel ilişkisiz oldukları deneysel olarak gösterilmiştir. Bu özellikten ve özilişki katsayılarının olasılık yoğunluk fonksiyonundaki dağılım parametreleri üzerinde yapılan varsayımlardan yararlanarak yeni stokastik kontrol algoritması yardımıyla öngörülen süzgeç katsayı vektörünün optimum katsayı vektörü için yansız bir kestireç olduğu analitik olarak gösterilmiştir. Önerilen yeni algoritma için, RLS algoritmasında kullanılan benzer işlemlerle, durağan olmayan ortamlarda da işleyen bir yapı elde edilmiştir. Yeni algoritmanın LMS ve RLS algoritmalarıyla yakınsama hızı ve madpr karşılaştırmaları yapılmıştır. Sistem tanılama uygulaması birinci ve ikinci dereceden doğrusal sistemler üzerinde pratik olarak gerçeklenmiştir.

SUMMARY

A NEW STOCHASTIC CONTROL ALGORITHM FOR ADAPTIVE FILTERING AND SYSTEM IDENTIFICATION APPLICATION

The design of an adaptive filter requires a priori information about the statistics of the data to be processed. The filter is optimum only when the statistical characteristic of the input data match the a priori information on which the design of the filter is based. When this information is not known completely , however , it may not be possible to design the filter or else the design may not be optimum. A straightforward approach is estimating the statistical parameters of the input signal. This is a two stage process whereby the filter first estimates the statistical parameters of the relevant signals and then uses them into a nonrecursive formula for computing the filter parameters. For real time operation , this procedure has the disadvantage of requiring excessively elaborate and costly hardware. A more efficient method is to use an adaptive control algorithm to update the filter parametres. The algorithm starts from some predetermined set of initial conditions , representing complete ignorance about the environment. In a stationary environment , we find that after succesive iterations of the algorithm it converges to the optimum solution in some statistical sense. In a nonstationary environment , the algorithm offers a tracking capability , whereby it can track time variations in the statistic of the input data , provided that the variations are sufficiently slow. As a direct consequence of the application of a recursive algorithm the parametres of an adaptive filter are updated from one iteration to the next they become data dependent.

The operation of an adaptive filtering algorithm involves two basic processes:

- Filtering process.
- Adaptive process.

The filtering process designed to produce an output in response to a sequence of input data. The purpose of adaptive process is to provide a mechanism for the adaptive control of an adjustable set of parametres used in filtering process. These two processes work interactively with each other. Naturally , the choise of a sturucture for the filtering process has a profound effect on the operation of the algorithm as a whole. The well known stuructures of an adaptive filter with finite memory or , equivalently finite impuls response (FIR) , are transversal filter and lattice filter. The transversal filter consists of three basic elements , unit delay element , multiplier and adder. The number of delay elements is commonly referred to as the order of the filter. For the transversal filter , the filter output y(n) is given by

$$y(n) = \sum_{k=0}^{M-1} w_k \, x(n-k)$$

Where x(n) is the filter input , w_k is the k.th filter parameter and M is the order of the filter.

Adaptive control algorithms which are used in the transversal adaptive filters can be classified into two main groups [1],[2].

-Gradient based algorithms.
-Recursive least squares algorithms.

Gradient based algorithms requires the use of a gradient vector , the value of which depends on two parametres: the correlation matrix **R** of the tap inputs x(n) , x(n-1) , ... , x(n-M+1) in the transversal filter and cross-correlation vector **p** between the desire response d(n) and the same tap inputs. Using instantaneous values for these correlations , an estimate can be obtained for the gradient vector. The resulting algorithm is widely known as the least mean square (LMS) or *stochastic gradient* algorithm [3],[4],[5]. This algorithm is simple and yet capable of achieving satisfactory performance under the rigth conditions. Its major limitations are relatively slow rate of convergence and a sensitivity to variations in the condition number of the correlations matrix (rate of the maximum eigenvalue to minimum eigenvalue) of the tap inputs. Nevertheless , the LMS algorithm is highly popular and widely used in variety of applications. The LMS algorithm can be written as follows:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu \, \mathbf{x}(n) \, [\, d(n) - \mathbf{x}^T(n) \, \mathbf{w}(n) \,]$$

Where , the index of n denotes discrete time variable , **x**(n) is the filter input vector with M dimensions , **w**(n) is the filter parameter vector and μ is the step-size parameter. The LMS algorithm converges to optimum solution $\mathbf{w}_o$

$$\mathbf{w}_o = \mathbf{R}^{-1} \, \mathbf{p}$$

with following condition :

$$0 < \mu < 2 / \lambda_{max}$$

where λ_{max} is the largest eigenvalue of the correlatin matrix **R**.

The derivation of the recursive least square (RLS) algorithm relies on a basic result in linear algebra known as the matrix inversion lemma [6],[7],[8]. An important feature of the RLS algorithm is that it utilizes information contained in the input data , extending back to the instant of time when the algorithm is initiated. The resulting rate of convergence is therefore typically an order of magnitude faster than the simple LMS algorithm [9],[10],[11],[12]. This inprovement in performance , however , is achieved at the expense of a large increase in computational complexity [13],[14],[15]. The RLS algorithm can be written as follows:

$$\mathbf{w}(n) = \mathbf{w}(n-1) + \mathbf{k}(n)\,\alpha(n)$$

where ,

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

$$\mathbf{R}^{-1}(n) = \mathbf{R}^{-1}(n-1) - \mathbf{k}(n)\,\mathbf{x}^{T}(n)\,\mathbf{R}^{-1}(n-1)$$

$$\alpha(n) = d(n) - \mathbf{w}^{T}(n-1)\,\mathbf{x}(n)$$

The standart RLS algorithm have a computational complexity (the number of multiplication , division in per recursion) that increases as the square of M , where M is the order of filter or number of adjustable weights in the algorithm. Such algorithms are often referred to as $O(M^2)$ algorithms , where $O(.)$ denotes order of. By contrast , in the LMS algorithm , its computational complexity increases linearly with M. The LMS is an $O(M)$ algorithm. When M is large , the computational complexity of the $O(M^2)$ algorithms may become objectionable from a hardware implementation point of wiev. There is therefore a motivation to modify the formulation of RLS algorithms in such a way that the computational complexity reduce to an $O(M)$ form. Resulting algorithms are known as fast algorithms that combine desirable characteristics of RLS with an $O(M)$ computational complexity. The disadventage of fast algorithms is that they needs complex hardware or software stuructures.

In this study , a new stochastic control algorithm is proposed. As well known , the optimum parameters of an adaptive filter are obtained from Wiener-Hopf equations as follows:

$$\mathbf{Rw} = \mathbf{p}$$

The original idea of this study is based on the iterative solution methods for linear systems [16],[17],[18],[19],[20]. The simplest iterative scheme is the Jacobi iteration. It is defined for matrices that have nonzero diagonal elements. The i.th parameter of an adaptive filter can be calculated by using the Jacobi method for the Wiener-Hopf equations , iteratively , as follows:

$$w_i\ (k+1) \;=\; \Big(p_i \;-\; \sum_{j=1}^{i-1} r_{i-j}\ w_j(k) \;-\; \sum_{j=i+1}^{M} r_{j-i}\ w_j(k) \Big) / r_0 \qquad i=1,\ldots,M$$

where p_i is the i.th element of cross-correlation vector $\mathbf{p}$ and r_{i-j} is the i.th row , j.th column element of correlation matrix $\mathbf{R}$. For the Jacobi iteration , the transition from $\mathbf{w}(k)$ to $\mathbf{w}(k+1)$ can be sufficiently described in terms of the lower triangular correlation matrix $\mathbf{R}_L$, upper triangular correlation matrix $\mathbf{R}_U$ and diagonal correlation matrix $\mathbf{R}_D$.

$$\mathbf{w}(k+1) \;=\; -\,\mathbf{R}_D^{-1}\,(\,\mathbf{R}_L \;+\mathbf{R}_U\,)\,\mathbf{w}(k) \;+\; \mathbf{R}_D^{-1}\ \mathbf{p}$$

The Jacobi algorithm converges to optimum solution w_0 under the condition

$$0 < 1/r_0 < 2/\lambda_{max}$$

Where r_0 is the variance of the input signal. This condition is not satisfied for every correlation matrix. For this reason the Jacobi algorithm needs an available step size parameter. The convergence rate of the Jacobi algorithm depends on the condition number of the correlation matrix.

Note that in the Jacobi algorithm one does not use the most recently available information when computing $w_i\ (k+1)$. For example , $w_1\ (k)$ is used in the calculation of $w_2\ (k+1)$ even though $w_1\ (k+1)$ is known. If we revise the Jacobi iteration so that we always use the most current estimate of the exact w_i , we obtain

$$w_i\ (k+1) \;=\; \Big(p_i \;-\; \sum_{j=1}^{i-1} r_{i-j}\ w_j(k+1) \;-\; \sum_{j=i+1}^{M} r_{j-i}\ w_j(k) \Big) / r_0 \qquad i=1,\ldots,M$$

This defines what is called the Gauss-Seidel algorithm for Wiener-Hopf equations. In matrix form , using proporties of $\mathbf{R}_U = \mathbf{R}_L^{\,T}$, Gauss-Seidel algorithm can be written as follows

$$\mathbf{w}(k+1) \;=\; -\;(\,\mathbf{R}_L \;+\mathbf{R}_D\,)^{-1}\,\mathbf{R}_L^{\,T}\,\mathbf{w}(k) \;+\; (\,\mathbf{R}_L \;+\mathbf{R}_D\,)^{-1}\,\mathbf{p}$$

It can be shown that the Gauss-Seidel algorithm always converges for positive definite symmetric matrices. The correlation matrix **R** is also positive definite symmetric matrix. So that , the Gauss-Seidel algorithm for Wiener-Hopf equations converges to optimum solution w_0 without any condition.

When the information about correlation matrix and cross correlation vector is not known completely , it may not be possible to use the Jacobi and Gauss-Seidel algorithms. A straightforward approach estimating the correlation matrix and cross correlation vector. The available estimates of correlation matrix and cross correlation vector are

$$\mathbf{R}(n) = 1/n \sum_{k=1}^{n} \mathbf{x}(k).\mathbf{x}^T(k)$$

$$\mathbf{p}(n) = 1/n \sum_{k=1}^{n} \mathbf{x}(k).d(k)$$

where n denotes time variable. Using estimates $\mathbf{R}(n)$ and $\mathbf{p}(n)$, the Jacobi and Gauss-Seidel algorithms can be rewritten as follows

$$\mathbf{w}(n+1) = - \mathbf{R}_D^{-1}(n)(\mathbf{R}_L(n) + \mathbf{R}_U(n)) \mathbf{w}(n) + \mathbf{R}_D^{-1}(n) \mathbf{p}(n)$$

$$\mathbf{w}(n+1) = - (\mathbf{R}_L(n) + \mathbf{R}_D(n))^{-1} \mathbf{R}_L^T \mathbf{w}(n) + (\mathbf{R}_L(n) + \mathbf{R}_D(n))^{-1} \mathbf{p}(n)$$

These algorithms can be addressed as *Stochastic Jacobi* and *Stochastic Gauss-Seidel* algorithms. Stochastic Jacobi algorithm converges in the mean sense under the condition which the Jacobi algorithm converges. It can be shown that the stochastic Gauss-Seidel algorithm always converges in the mean sense by using distribution of correlation coefficients and some assumptions about on parametres of distribution [21]. So , we can say that the estimate of filter parameter vector $\mathbf{w}(n)$ which is obtained by using the stochastic Gauss-Seidel algorithm is an *unbiased* estimater of optimum solution $\mathbf{w}_0$.

$$E[\ \mathbf{w}(n)\ ] = \mathbf{w}_0$$

Where E[.] denotes the expected value . Altough the stochastic Gauss-Seidel algorithm does not need a condition about the eigenvalues of the correlation matrix to convergence , its convergence rate depends on the condition number of the correlation matrix. The studies on the numerical examples which consists of correlation matrices with various condition numbers shows that the convergence rate of the stochastic Gauss-Seidel algorithm is faster than the LMS algorithms rate.

The stochastic Gauss-Seidel algorithm is an $O(M^2)$ algorithm. The number of multiplication and division per recursion (*madpr*) in the stochastic Gauss-Seidel algorithm increases with M^2 . In generally , the relation between *madpr* and order of filter M in the stochastic Gauss-Seidel algorithm is as follows

$$madpr = M^2 + 2M - 1$$

In RLS algorithm *madpr* is equal to $3M^2 + 11M + 8$. It is clear that the computational comlexity of the stochastic Gauss-Seidel algorithm is better than the RLS algorithm. In fast algorithms (Fast RLS) *madpr* is equal to 7M +16 [22],[23],[24],[25],[26]. It can be seen that , altough , the stochastic Gauss-Seidel algorithm is an $O(M^2)$ algorithm , the computational comlexity of the stochastic Gauss-Seidel algorithm is better than the Fast RLS algorithm up to M=7.

In the application of system identification of adaptive filter [27],[28],[29],[30] , especially in the example of this study , the *madpr* of the stochastic Gauss-Seidel algorithm decreases. So that , the *madpr* is equal to $M^2 + M$. In such a case the stochastic Gauss-Seidel algorithm can be compaired to Fast RLS algorithm up to M=8.

BÖLÜM 1

GİRİŞ

Sistem tanılama (identification), işaret modelleme, öngörü (prediction) gibi belli başlı uygulama alanları olan uyarlamalı süzgeçler (adaptive filter) belirli bir süzme işlevinin yerine getirilmesi amacıyla uygun bir kontrol algoritması ile uyarılan yapılardır.

Uyarlamalı süzgeçlerde karşılaşılan süzgeç yapıları, sonlu dürtü cevaplı doğrusal süzgeçler (linear FIR filter) ve kafes (lattice) süzgeçlerdir. Doğrusal FIR süzgeç içeren uyarlamalı işaret işlemede kullanılan kontrol algoritmaları iki ana sınıfta toplanabilir :

- Gradient tabanlı algoritmalar.
- Ardışıl en küçük kareler algoritması ve türevleri.

Gradient tabanlı algoritmalar istenen amaç ölçütünü en dik düşüm (steepest descent) yöntemi ile minimum yapan algoritmalardır. En yaygın olarak kullanılan ve bilinen gradient tabanlı uyarlamalı algoritma en küçük karesel ortalama (LMS, least- mean-square) algoritmasıdır. Stokastik gradient algoritması olarak da adlandırılan LMS ikinci bölümde incelenecektir.

Ardışıl algoritmaların temelini RLS (recursive least square) oluşturmaktadır. RLS in çalışma ilkesi, ilgilenilen stokastik sürece ait olan özilişki matrisinin (correlation matrix) tersinin her adımda ardışıl olarak hesaplanmasına dayanır. Hızlı ardışıl algoritmalarda (Fast RLS) amaç bir iterasyon adımındaki işlem sayısını azaltmaktır. RLS ve özellikleri ikinci bölümde incelenecektir.

Her iki sınıf kontrol algoritmasında da minimize edilmek istenen amaç ölçütü aynıdır. Amaç ölçütünün yapısını, doğrusal FIR süzgeç

yapısı, süzgeç giriş ve çıkış işareti belirler. Süzgeç yapısının incelenmesi ve amaç ölçütünün tanımlanması ikinci bölümde verlmiştir. Hata ölçütü en genel durumda, süzgeç giriş işareti ile çıkış işaretinin farkından oluşan hata teriminin karesel ortalamasıdır. Hata teriminin tanımı uyarlamalı süzgecin uygulama alanına göre değişir. Örneğin sistem tanılama uygulamasında hata, süzgeç çıkışı ile parametreleri kestirilmek istenen sistemin çıkışı arasındaki farktır.

Kontrol algoritmalarının birbirleriyle karşılaştırılmasına olanak veren ölçütlerin önemlilerinden biri yakınsama hızıdır. Yakınsama hızı algoritmanın optimum çözüme ulaşması için gerekli olan iterasyon sayısıyla ilişkilidir. İterasyon sayısı küçüldükçe algoritmanın hızının artacağı açıktır. Doğal olarak algoritmalardan istenen yüksek bir yakınsama hızıdır. Gradient tabanlı algoritmalarda , yakınsama için bazı koşulların gerçeklenmesi gerekebilir. Bu koşulların kolay gerçeklenebilir olması veya mümkünse algoritmanın koşul gerektirmeden yakınsaması özelliği aranır. Stokastik sürecin özilişki matrisinin yapısı gradient tabanlı algoritmaların yakınsama hızlarına ve yakınsama koşullarına etkir. Bu etki özilişki matrisinin, en büyük özdeğerinin en küçük özdeğerine oranını olarak tanımlanan *oransal özdeğer yaygınlığına* bağlıdır. Oransal özdeğer yaygınlığı büyüdükçe yakınsama hızı azalmaktadır.

Kontrol algoritmalarının birbiriyle karşılaştırılmasının ikinci bir ölçütü de her bir iterasyondaki işlem sayısıdır. İşlem denince toplama, çıkarma, çarpma, bölme gibi cebirsel işlemlerin yanında zamanda gecikme veya öteleme gibi süzgeç yapsıyla ilgili işlemlerde düşünülmelidir. Uyarlamalı süzgecin gerçeklenmesi ister donanım ister yazılım şeklinde olsun, bu işlemlerden maaliyeti yüksek olanlar çarpma ve bölme işlemleridir. Bu bakımdan bir iterasyondaki çarpma ve bölme sayısının küçük olması istenen bir özelliktir.

Bu çalışmada, uyarlamalı doğrusal FIR süzgeçlerde istenen amaç ölçütünü minimum yapmak amacıyla, yeni bir uyarlamalı kontrol algoritması önerilmiştir. Önerilen algoritmanın yakınsama koşulları

incelenmiş, bu koşulları iyileştimek amacıyla algoritmanın yapısında bazı değişiklikler gerçeklenmiştir. Yakınsama hızı ve koşulu bakımından LMS ile yapılan karşılaştırmada önerilen yeni algoritmanın daha üstün olduğu anlaşılmıştır. Hız ve hızın özilişki matrisinin yapısına bağımlılığı konusunda LMS e göre olan bu üstünlük, RLS e göre yapılan bir karşılaştırmada söz konusu değildir. Önerilen algoritmanın bir iterasyonundaki işlem sayısı incelendiğinde, bu sayının LMS e göre daha büyük, RLS ile yapılan karşılaştırmada daha küçük olduğu anlaşılmıştır. LMS de işlem sayısı, süzgeç uzunluğu ile doğrusal olarak artarken, RLS de bu artış karesel olmaktadır. Yeni önerilen algoritmada da işlem sayısı süzgeç uzunluğu ile karesel olarak büyüse de RLS algoritmasına göre daha yavaş olmaktadır. Bu çalışmada sunulan yeni yöntemin, hız ve işlem sayısı esas alınarak yapılan bir karşılaştırmada LMS ile RLS arasında olduğu söylenebilir.

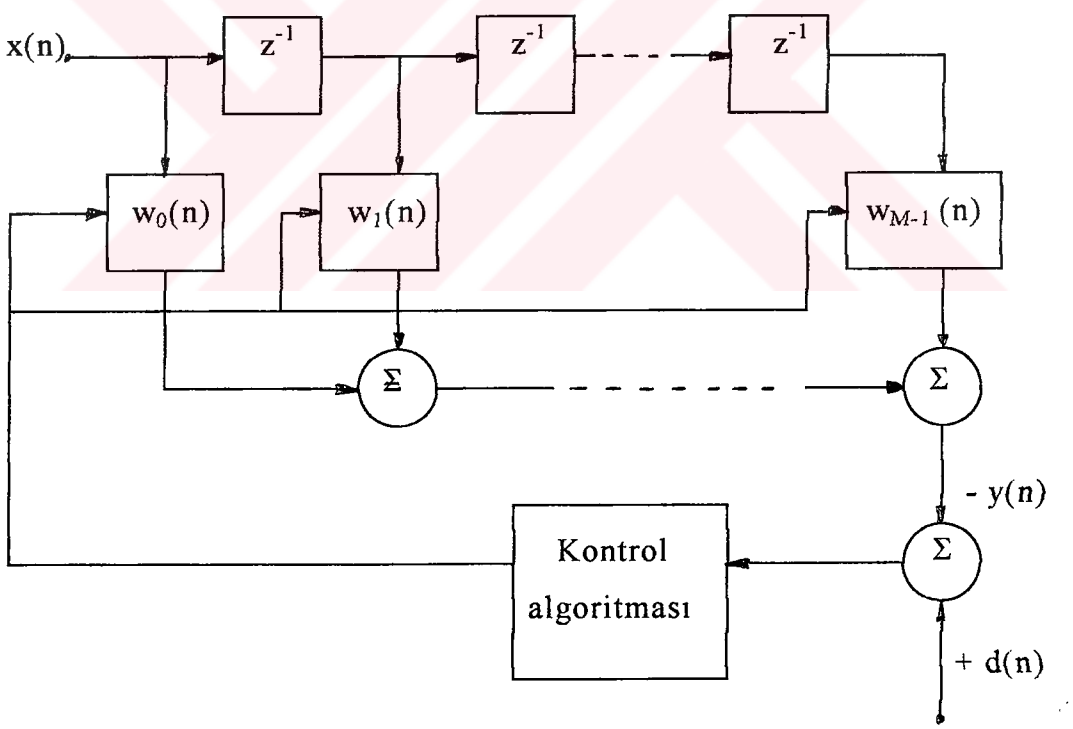
Yapılan çalışmada, uyarlamalı doğrusal FIR süzgeçler yardımıyla doğrusal sistem tanılama uygulaması pratik olarak gerçeklenmiştir. Beyaz gürültü kaynağı ile sürülen birinci ve ikinci dereceden doğrusal sistemlerin çıkışından alınan işaretler kullanılarak bu sistemlere ait parametre kestirimleri yapılmıştır. Örnekleme işlemi için PCL812 ADC kartı kullanılmıştır. LMS, RLS ve önerilen yeni kontrol algoritması bir PC de yazılım olarak gerçeklenmiştir. Gerçek sistemlerdeki elemanların ideal eleman olmaması , yüksek frekanslarda direnç elemanlarındaki kapasitif etki ve kapasite elemanlarındaki kayıp faktörü etkisi sonucu sistem transfer fonksiyonunda belirlenemeyen değişiklikler oluşmuştur. Bu olumsuzluklardan etkilenmemek amacıyla , sistem parametrelerinin değişik değerleri için algoritmaların yakınsama hızı karşılaştırmaları simülasyon sonuçları üzerinden yapılmıştır.

BÖLÜM 2

UYARLAMALI FIR SÜZGEÇLER VE KONTROL ALGORİTMALARI

2.1 Uyarlamalı FIR Süzgeç Yapısı

Uyarlamalı süzgeçlerde, süzgeç katsayıları uyarlama süreci boyunca kontrol algoritması tarafından, amaç ölçütünü minimuma indirecek şekilde değiştirilir. Şekil (2.1) den anlaşılacağı gibi kontrol algoritması ve FIR süzgeç katsayıları etkileşim halindedir.



Şekil 2.1 Uyarlamalı FIR Süzgeç

Süzgeç girişindeki x(n) işareti ortalaması sıfır olan durağan (istatistiksel özellikleri zamanla değişmeyen) bir sürece ait örneklerden oluşmaktadır. Süzgeç çıkışındaki y(n) işareti, giriş işareti x(n) in zaman domeninde ötelenmiş değerlerinin doğrusal bir fonksiyonudur. Kontrol algoritmasının girişindeki hata terimi e(n) süzgeç çıkışı y(n) ile d(n) nin farkından oluşmaktadır. Sistem tanılama uygulamalarında d(n), ilgilenilen doğrusal sistemin çıkışındaki işarettir. Uyarlamalı işaret işlemede amaç, hatanın karesel ortalamasını minimum yapacak şekilde, y(n) nin d(n) yi izlemesini sağlamaktır.

$$\mathbf{x}(n) = [x(n), x(n-1),. . . .,x(n-M+1)]^T \qquad (2.1.1)$$

$$\mathbf{w}(n) = [w_0(n), w_1(n), ...,w_{M-1}(n)]^T \qquad (2.1.2)$$

Süzgeç çıkış işareti y(n) , (2.1.1), (2.12) de tanımlanmış olan **x** giriş vektörü ve **w** süzgeç katsayı vektörü cinsinden

$$y(n) = \mathbf{x}(n)^T.\mathbf{w}(n) \qquad (2.1.3)$$

olarak yazılabilir. Hatanın karesel ortalaması, d(n) ve y(n) cinsinden

$$E[e(n)^2] = E[(d(n)-y(n))^2] \qquad (2.1.4)$$

elde edilir. Süzgeç çıkışı y(n) yerine (2.1.3) deki eşiti yazılır ve (2.1.4) yeniden düzenlenlenerek :

$$E[e(n)^2] = E[(d(n) - \mathbf{x}(n)^T.\mathbf{w}(n))^2] \qquad (2.1.5)$$

bulunur. Giriş işareti x(n) nin özilişki matrisi **R** ile , x(n) ile d(n) arasındaki çapraz ilişki vektörü **p** nin :

$$\mathbf{R} = E[\ \mathbf{x}(n).\ \mathbf{x}^T(n)\ ] \qquad (2.1.6)$$

$$\mathbf{p} = E[\, d(n).\mathbf{x}(n)] \qquad (2.1.7)$$

tanım bağıntıları kullanılarak hatanın karesel ortalaması :

$$E[e(n)^2] = \sigma_d^2 - \mathbf{w}^T(n).\mathbf{p} - \mathbf{p}^T\mathbf{w}(n) + \mathbf{w}^T(n).\mathbf{R}.\mathbf{w}(n) \qquad (2.1.8)$$

ifadesi elde edilir. (2.1.8) eşitliğindeki σ_d^2 , d(n) nin varyansıdır.

2.1.1 Optimum Süzgeç Katsayıları

Hatanın karesel ortalaması, (2.1.8) eşitliğinden anlaşılacağı gibi, $\mathbf{w}(n)$ katsayı vektörünün bir fonksiyonudur. Karesel ortalamayı minimum yapan $\mathbf{w}_0$ vektörünün elemanları optimum süzgeç katsayıları veya optimum Wiener çözümü olarak bilinir [1]. Bu çözümü elde etmek amacıyla (2.1.8) eştliği $\mathbf{w}$ nın vektörel bir fonksiyonu şeklinde düzenlenirse :

$$F(\mathbf{w}) = \sigma_d^2 - \mathbf{w}^T.\mathbf{p} - \mathbf{p}^T\mathbf{w} + \mathbf{w}^T.\mathbf{R}.\mathbf{w} \qquad (2.1.9)$$

eşitliği bulunur. $F(\mathbf{w})$ yı $\mathbf{w}$ ya göre minimum yapmak için (2.1.9) eşitliğinin her iki tarafına gradient operatörü uygulanırsa :

$$\nabla_w F(\mathbf{w}) = -2.\mathbf{p} + 2.\mathbf{R}.\mathbf{w} \qquad (2.1.10)$$

elde edilir. Optimum çözümü bulmak için (2.1.10) denklem takımı sıfır vektörüne eşitlenerek :

$$\mathbf{R}.\mathbf{w}_0 = \mathbf{p} \qquad (2.1.11)$$

sonucuna varılır. (2.1.11) deki doğrusal denklem takımı Wiener Hopf eştlikleri olarak adlandırılır [1].

2.2 Kontrol Algoritmaları

Uyarlamalı işaret işlemede amaç süzgeç işlemi süresince süzgeç katsayılarını optimum Wiener çözümüne yaklaştırmaktır. Bu işlem uyarlamalı kontrol algoritmaları ile sağlanır. Kontrol algoritmaları, (2.1.9) eşitliği ile verilmiş olan amaç ölçütünü minimum yapmak için kullanılan bazı optimizasyon teknikleri veya (2.1.11) eşitliğindeki Wiener-Hopf doğrusal denklem takımını ardışıl olarak çözen yöntemlerdir. Bu optimizayon tekniklerinden en çok kullanılan ve bilineni, gradient tabanlı bir yöntem olan en dik düşüm (steepest descent) yöntemidir [2].

2.2.1 En Dik Düşüm (Steepest Descent) Algoritması

Bu çalışmada önerilen uyarlamalı kontrol algoritmasının kararlılık ve yakınsama hızı analizlerinde en dik düşüm algoritmasının işlemlerine benzer işlemler uygulanmıştır. Bu nedenle, LMS in de temelini oluşturan steepest descent algoritmasının kararlılık ve yakınsama hızı incelemesine yer verilmesi uygun görülmüştür. Amaç ölçütü olarak F(**w**) vektörel fonksiyonunu kullanan en dik düşüm algoritması (2.2.1) eşitliği ile verilmiştir.

$$\mathbf{w}(k+1) = \mathbf{w}(k) - (1/2).\mu.\nabla_w F(\mathbf{w}(k)) \qquad (2.2.1)$$

Adım parametresi (step size parameter) olarak adlandırılan μ sabiti algoritmanın optimum çözüme yakınsama hızını etkiler. Algoritmanın yakınsyabilmesi için μ nün sağlaması gereken koşul ileride verilecektir..

F(**w**) nın (2.1.10) bağıntısında elde edilmiş olan gradienti (2.2.1) de yerine yazılarak :

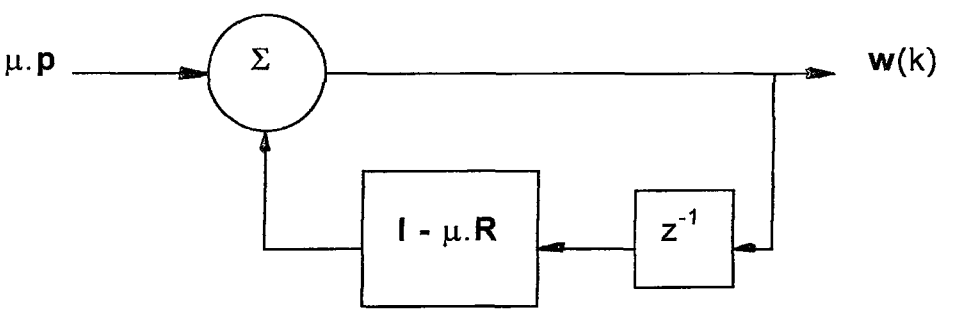
$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu.[\ \mathbf{p} - \mathbf{R}.\mathbf{w}(k)\ ]$$

$$\mathbf{w}(k+1) = \mu\ \mathbf{p} + [\ \mathbf{I} - \mu\mathbf{R}\ ].\mathbf{w}(k) \qquad (2.2.2)$$

matris formundaki fark denklemi elde edilir. **I**, **R** nin boyutundaki birim matristir.

2.2.2 Kararlılık İncelemesi

(2.2.2) eşitliği ile verilmiş olan matris formundaki fark denklemleri, girişi $\mu\mathbf{p}$, çıkışı $\mathbf{w}(k)$ olan çok giriş, çok çıkışlı birinci dereceden doğrusal bir sistemin modeli olarak yorumlanabilir. Bu sistemin blok şeması şekil (2.2) de verilmiştir. Sistemin kararlı olabilmesi başka bir deyişle $\mathbf{w}(k)$ nın optimum Wiener çözümüne yakınsaması için adım parametresi μ nün sağlaması gereken bir koşul vardır. Sistemin kararlılık incelemesi amacıyla (2.2.2) deki fark denklemi çözülerek :



Şekil 2.2

$$\mathbf{w}(k) = (\mathbf{I} - \mu.\mathbf{R})^k.\mathbf{w}(0) + [\sum_{n=0}^{k-1} (\mathbf{I} - \mu.\mathbf{R})^n].\mu.\mathbf{p} \qquad (2.2.3)$$

elde edilir. (2.2.3) deki dizinin yakınsak olması için (**I** - μ.**R**) matrisinin en büyük özdeğerinin modülünün birden küçük olması gerekir. Bu koşulun sağlandığını görebilmek amacıyla **R** matrisinin özelliklerinden yararlanılabilir. **R** matrisi pozitif tanımlı simetrik bir matristir [1]. Bu özellik kullanılarak **R** matrisi :

$$\mathbf{R} = \mathbf{Q}\Lambda\mathbf{Q}^T \qquad (2.2.4)$$

şeklinde yazılabilir. (2.2.4) eşitliğinde Λ, özilişki matrisi **R** nin özdeğerlerinden oluşan diagonal matristir. **Q** matrisinin sütunları **R** nin özvektörleridir.

$$\mathbf{Q}\mathbf{Q}^T = \mathbf{I} \qquad (2.2.5)$$

özelliği kullanılarak **R** nin k. kuvvetinin :

$$\mathbf{R}^k = \mathbf{Q}\,\Lambda^k\,\mathbf{Q}^T \qquad (2.2.6)$$

olduğu görülebilir. Benzer işlemler yapılarak :

$$\mathbf{Q}^T (\mathbf{I} - \mu.\mathbf{R})^k \mathbf{Q} = (\mathbf{I} - \mu.\Lambda)^k \qquad (2.2.7)$$

sonucuna varılır. (2.2.7) eşitliğinden anlaşılacağı gibi $(\mathbf{I} - \mu.\mathbf{R})^k$ matrisinin k değeri sonsuza giderken sonlu kalabilmesi için $(\mathbf{I} - \mu.\Lambda)^k$ matrisinin elemanlarının sonlu kalması gerekmektedir. Bu durumda λ_i , **R** nin herhangi bir özdeğeri olmak üzere her i için :

$$|1 - \mu.\lambda_i| < 1 \qquad (2.2.8)$$

koşulunun sağlanması gerekir. (2.2.6) eşitsizliği düzenlenerek :

$$0 < \mu.\lambda_i < 2$$

$$0 < \mu < 2 / \lambda_{max} \qquad (2.2.9)$$

şeklinde yazılabilir. (2.2.9) eşitsizliğindeki koşul sağlanırsa (2.2.3) deki $\mathbf{w}(k)$ vektörü k nın değeri sonsuza giderken :

$$\mathbf{w}(k) = \mathbf{R}^{-1}.\mathbf{p} \qquad (2.2.10)$$

olur. Bu ifade (2.1.11) eşitliği ile verilmiş olan optimum Wiener çözümüdür.

2.2.3 Yakınsama Hızı İncelemesi

Steepest descent algoritmasının optimum Wiener çözümüne yakınsaması için gerekli iterasyon sayısı, adım parametresi μ nün değeri ve özilişki matrisi $\mathbf{R}$ nin oransal özdeğer yaygınlığı ile ilgilidir. μ nün yakınsama hızına etkisini incelemek amacıyla :

$$\mathbf{e}_w(k) = \mathbf{w}(k) - \mathbf{w}_0$$

$$\mathbf{e}_w(k+1) = \mathbf{w}(k+1) - \mathbf{w}_0 \qquad (2.2.11)$$

şeklinde süzgeç katsayıları hata vektörü tanımlansın. (2.1.11), (2.2.11) ve (2.2.2) eştliklerinden yararlanılarak :

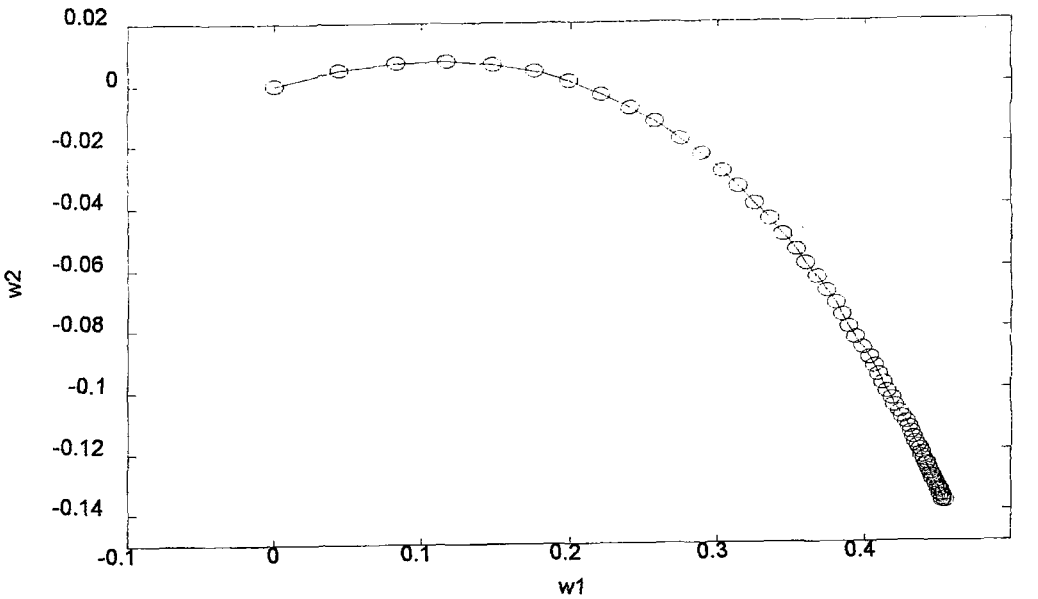
$$\mathbf{e}_w(k+1) = (\mathbf{I} - \mu.\mathbf{R}).\mathbf{e}_w(k) \qquad (2.2.12)$$

elde edilir. (2.2.12) deki fark denklemi çözülerek :

$$\mathbf{e}_w(k) = (\mathbf{I} - \mu.\mathbf{R})^k \, \mathbf{e}_w(0) \qquad (2.2.14)$$

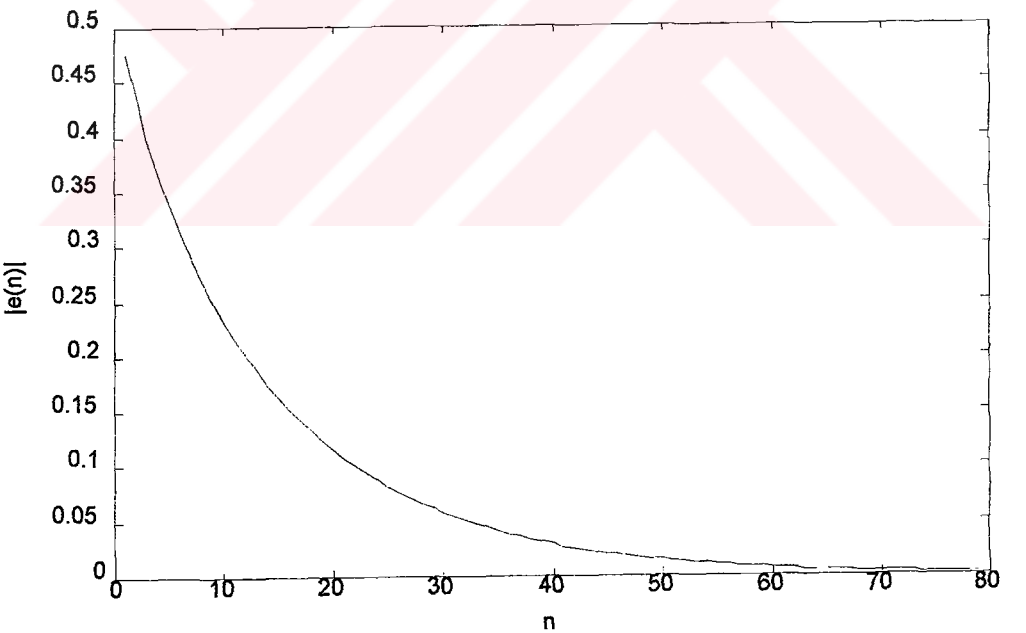
sonucuna varılır. (2.2.14) ve (2.2.7) eşitlikleri incelenerek hata vektörünün sıfır vektörü olması için $(\mathbf{I} - \mu.\Lambda)^k$ matrisinin k büyüdükçe sıfır matrisine yaklaşması gerektiği anlaşılır. Λ matrisinin **R** nin özdeğerlerinden oluşan diagonal bir matris olduğu düşünülürse, $(\mathbf{I}-\mu.\Lambda)^k$ matrisinin i. elemanının $(1- \mu.\lambda_i)^k$ olacağı görülür. (2.2.9) ile belirtilmiş eşitsizliğin sağlanması koşuluyla μ nün değeri hata vektörünün sıfır vektörüne yaklaşma hızını etkiliyecektir. Özilişki matrisinin yapısı da yakınsama hızını etkiliyen başka bir faktödür.

Oransal özdeğer yaygınlığı ve μ nün çeşitli değerleri için , iki boyutlu durumda $\mathbf{w}(k)$ vektörünün optimum Wiener çözümüne yakınsaması ve hata vektörü $\mathbf{e}_w(k)$ nın normunun sıfıra yaklaşması şekil (2.3) , (2.4) , (2.5) , (2.6) , (2.7) , (2.8) , (2.9) ve (2.10) de verilmiştir. Oransal özdeğer yaygınlığının ve bununla ilişkili olan optimum katsayı vektörlerinin sayısal değerleri [1] numaralı kaynaktaki bir örnekten alınmıştır. Oransal özdeğer yaygınlığının küçük olduğu durumlarda optimum çözüme daha çabuk ulaşılmaktadır. Aynı oransal özdeğer yaygınlığı için yakınsama hızı adım parametresi μ ile değişmektedir. μ nün küçük değerleri için yakınsama yavaş olmaktadır. Özdeğer saçılımının küçük olduğu durumda μ=0.1 için hata vektörünün normu yaklaşık 70-80 adımda sıfıra yaklaşmaktadır. Bu durum şekil (2.4) de görülmektedir. Oransal özdeğer yaygınlığının daha büyük olduğu durmda μ=0.1için hata vektörünün normunun sıfır olması yaklaşık 100 adımda sağlanmaktadır. Bu durum şekil (2.8) da verilmiştir.



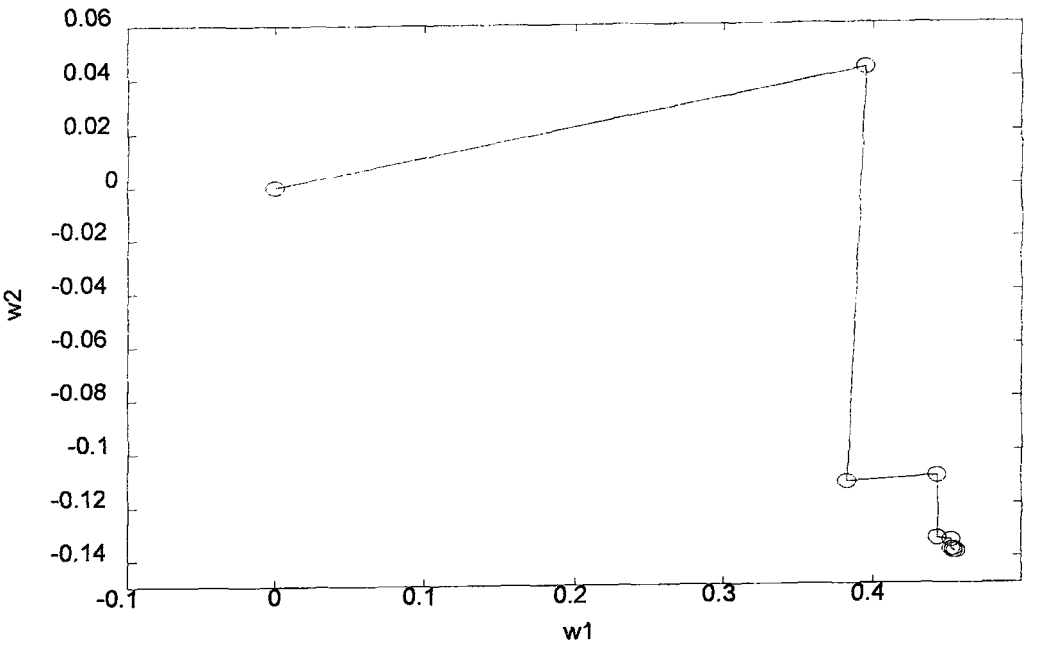
Şekil 2.3

$w_0 = [\ 0.4553 \quad -0.1381\ ]$, $\lambda_{max}/\lambda_{min} = 6$, $\mu = 0.1$



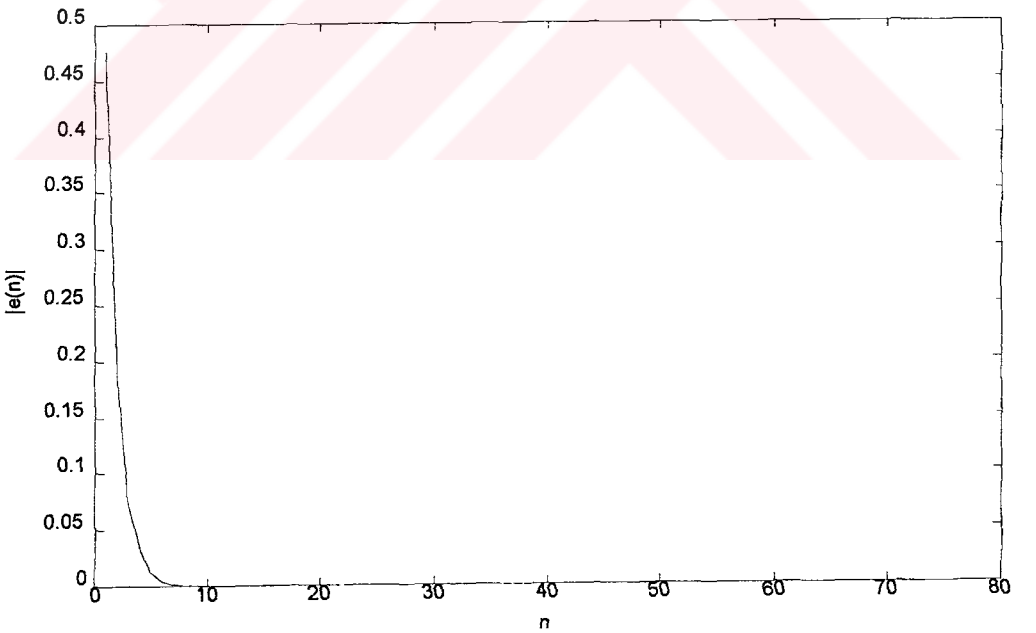
Şekil 2.4

$w_0 = [\ 0.4553 \quad -0.1381\ ]$, $\lambda_{max}/\lambda_{min} = 6$, $\mu = 0.1$



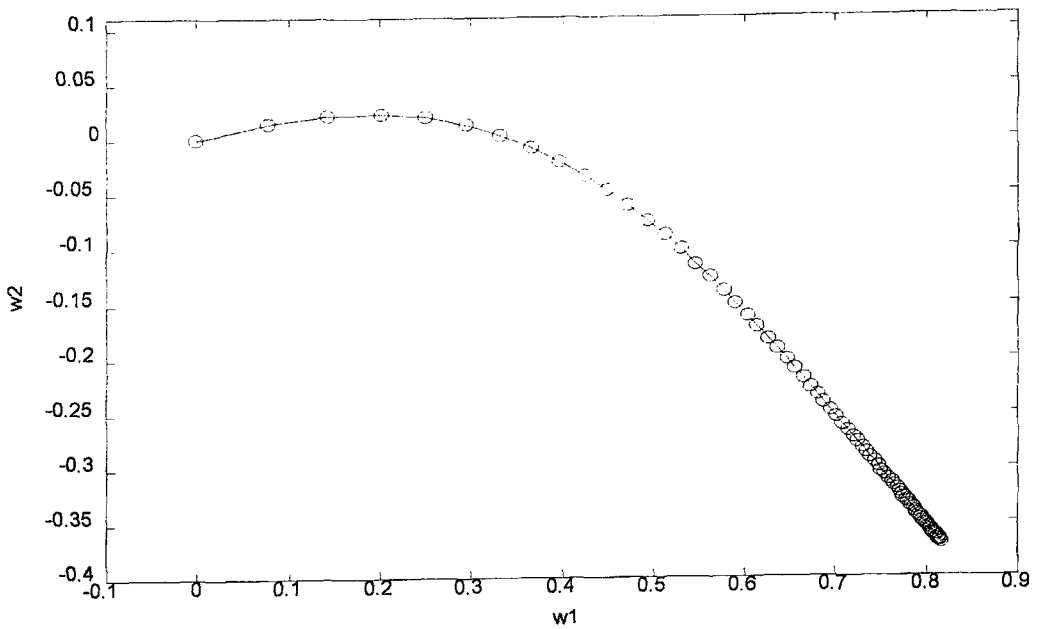
Şekil 2.5

$w_0 = [\ 0.4553 \quad -0.1381\ ]$, $\lambda_{max}/\lambda_{min} = 6$, $\mu = 0.9$



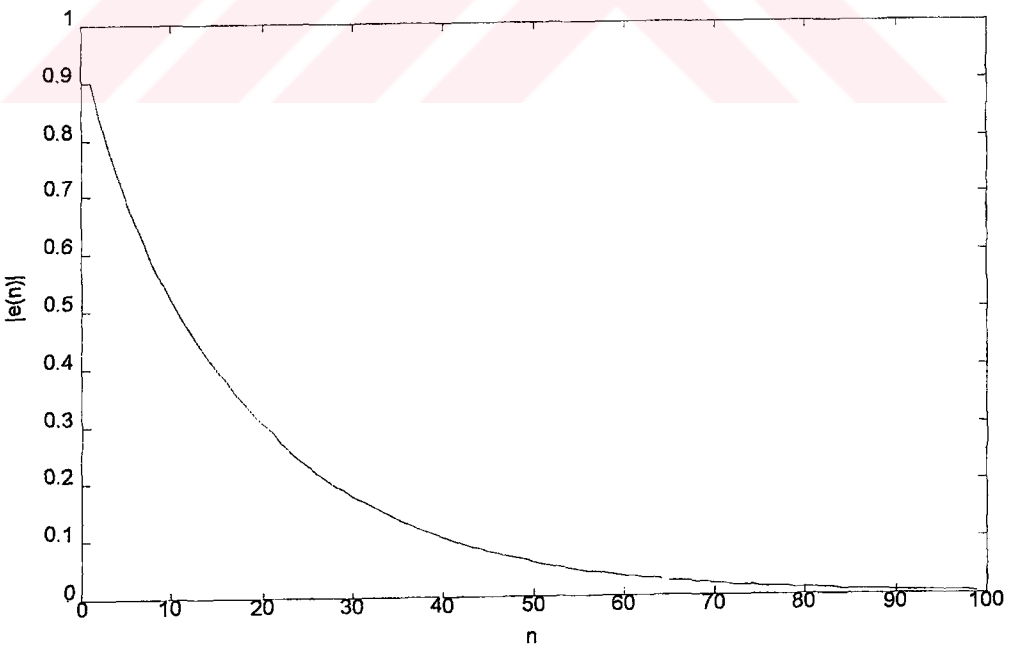
Şekil 2.6

$w_0 = [\ 0.4553 \quad -0.1381\ ]$, $\lambda_{max}/\lambda_{min} = 6$, $\mu = 0.9$



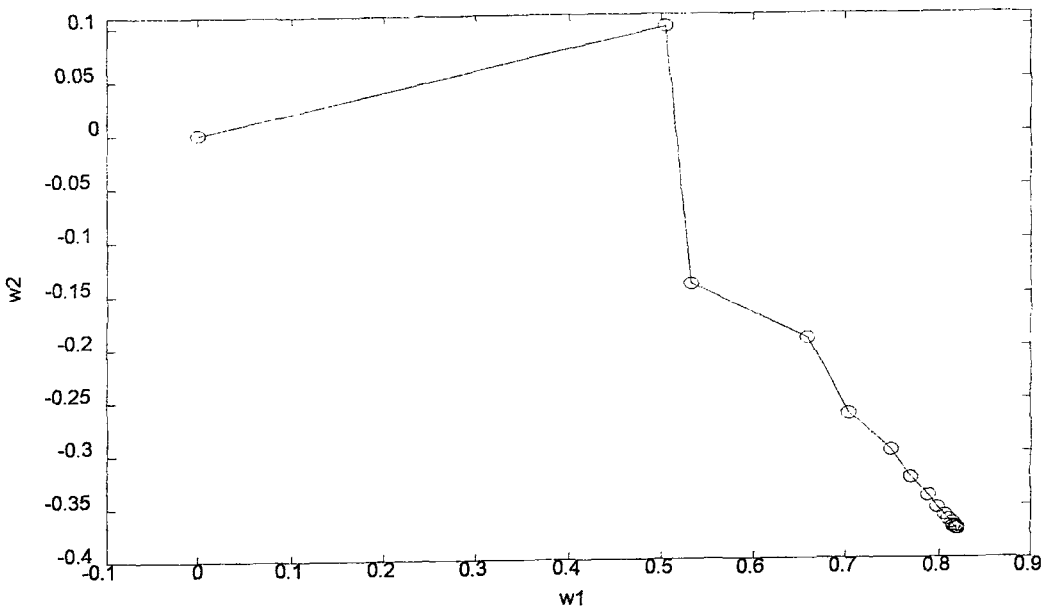
Şekil 2.7

$w_0 = [\ 0.8197 \quad -0.3733\ ]$, $\lambda_{max}/\lambda_{min} = 46.8$, $\mu = 0.1$



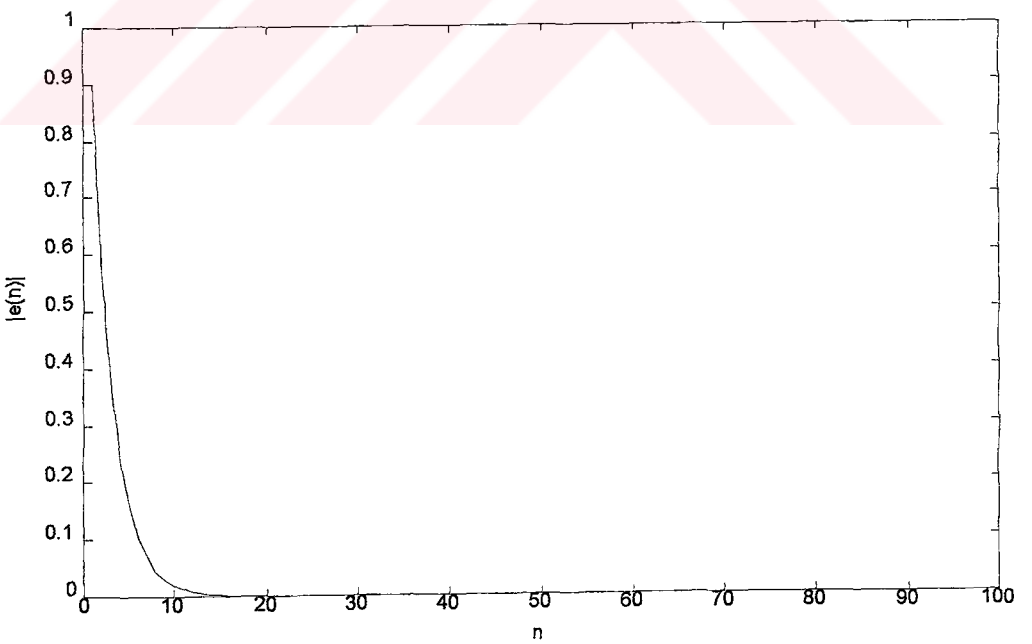
Şekil 2.8

$w_0 = [\ 0.8197 \quad -0.3733\ ]$, $\lambda_{max}/\lambda_{min} = 46.8$, $\mu = 0.1$



Şekil 2.9

w_0 = [0.8197 -0.3733] , $\lambda_{max}/\lambda_{min}$ = 46.8 , μ = 0.65



Şekil 2.10

w_0 = [0.8197 -0.3733] , $\lambda_{max}/\lambda_{min}$ = 46.8 , μ = 0.65

2.3 Uyarlamalı Algoritmalar

Özilişki matrisi R ve çapraz ilişki vektörü p nin tam olarak bilinmesi durumunda optimum Wiener çözümünü elde etmek için R matrisinin tersini almak ve R^{-1} matrisini p ile çarpmak yeterli olacaktır. Bu işlem için bir algoritma kullanma gereği de yoktur. Ancak R ve p nin tam olarak elde edilebilmesi için ilgilenilen stokastik süreçler x(n) ve d(n) üzerinde uzun süreli gözlem yapılması gerekmektedir. Bu gözlem sonunda elde edilen örnekler kullanılarak optimum Wiener çözümünün hesaplanması bir uyarlama işlemi olarak adlandırılamaz. Uyarlamalı işaret işlemede amaç x(n) ve d(n) süreçlerinden örnekler geldikçe her adımda w katsayı vektörünün yeni duruma uyarlanarak optimum çözüme yaklaşmasını sağlamaktır. Bu uyarlama işlemi süresinde R ve p nin öngörülmüş değerleri kullanılır. Belli bir örnek kümesi için en uygun öngörü değerleri :

$$\mathbf{R}_{ort} = 1/N \sum_{k=1}^{N} \mathbf{x}(k).\mathbf{x}^{T}(k) \qquad (2.3.1)$$

$$\mathbf{p}_{ort} = 1/N \sum_{k=1}^{N} \mathbf{x}(k).d(k) \qquad (2.3.2)$$

eşitlikleri ile verilen örnek ortalamalarıdır. R ve p nin gerçek değerlerini yerine (2.3.1) ve (2.3.2) ile verilen ortalama değerleri kullanılarak optimum Wiener çözümünün ardışıl olarak bulunması ardışıl enküçük kareler algoritmasının (RLS) temelini oluşturur. Özilişki matrisi R ve çapraz ilişki vektörü p yerine bunların n anındaki ani değerlerinin kullanıldığı gradient tabanlı uyarlamalı algoritma LMS , enküçük karesel ortalama algoritması olarak bilinmektedir. R ve p nin n anındaki ani değerleri :

$$\mathbf{R}_{ani}(n) = \mathbf{x}(n).\mathbf{x}^T(n) \qquad (2.3.3)$$

$$\mathbf{p}_{ani}(n) = \mathbf{x}(n).d(n) \qquad (2.3.4)$$

eşitlikleriyle verilmiştir.

2.3.1 Enküçük Karesel Ortalama (LMS) Algoritması

Stokastik gradient algoritması olarak da bilinen LMS in temeli ilgilenilen amaç ölçütünün (2.1.10) eşitliği ile verilen gradient vektörü yerine bu gradient vektörün n anındaki ani değerinin kullanılması ilkesine dayanır. Gradient vektörün ani değerinin elde edilmesi için (2.3.3) ve (2.3.4) eşitliklerindeki R ve p nin ani değerleri kullanılır. Bu durumda gradient vektörü :

$$\nabla_w(n) = -2.\mathbf{p}_{ani}(n) + 2.\mathbf{R}_{ani}(n).\mathbf{w}(n) \qquad (2.3.5)$$

$$\nabla_w(n) = -2.\mathbf{x}(n).d(n) + 2.\mathbf{x}(n).\mathbf{x}^T(n).\mathbf{w}(n) \qquad (2.3.6)$$

olarak bulunur. (2.3.6) eşitliğindeki gradient vektörün ani değeri (2.2.1) ile verilmiş olan algoritmada yerine yazılırsa :

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu.\mathbf{x}(n).[\, d(n) - \mathbf{x}^T(n).\mathbf{w}(n) \,] \qquad (2.3.7)$$

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu.\mathbf{x}(n).e(n) \qquad (2.3.8)$$

(2.3.7) veya (2.3.8) ile tanımlanan LMS algoritması elde edilir [3],[4],[5].

LMS in en önemli üstünlüğü programlamaya uygun basit yapısı ve bir iterasyondaki işlem sayısının küçük olmasıdır. Süzgeç uzunluğu veya

w katsayı vektörü boyutu M olan bir uyarlama işleminde n anındaki hata e(n) i hesaplamak için iki tane M boyutlu vektörün çarpılması gerekir. Bu hesaplama işlemi M tane çarpma ve M tane toplama demektir. (2.3.8) eşitliğinde sağdaki ikinci terimi elde etmek için de M tane çarpma işlemi gerekir. Katsayı vektörü w(n) nin bir sonraki adımdaki değerine uyarlanması için toplam 2M çarpma ve 2M toplama işlemi gerekmektedir. Bir iterasyon adımındaki çarpma ve bölme (multiplication and division per recursion) sayısı *madpr* olarak adlandırılmaktadır [6]. LMS algoritması için

$$(madpr)_{\mathrm{LMS}} = 2\mathrm{M}$$

olmaktadır. Görüldüğü gibi LMS de madpr süzgeç uzunluğu M ile doğrusal olarak artmaktadır.

2.3.1.1 Kararlılık İncelemesi

Gradient vektörünün (2.3.6) eşitliği ile verilmiş olan ifadesindeki $\mathbf{x}(n)$ vektörünün elemanları ve d(n) işareti stokastik birer süreç oldukları için gradient vektörünün n anındaki değeri $\nabla_w(n)$ bir raslantı vektörü olacaktır. Bunun doğal sonucu olarak (2.3.8) eşitliğinde tanımlanan $\mathbf{w}(n)$ katsayı vektörü de bir raslantı vektörüdür. Kararlılık incelemesinin $\mathbf{w}(n)$ için değil bu raslantı vektörünün ortalama değeri için yapılması uygun olur. (2.2.11) de verilmiş olan katsayı hata vektörü $\mathbf{e}_w(k)$ ya benzer şekilde tanımlanan :

$$\mathbf{e}_w(n) = \mathbf{w}(n) - \mathbf{w}_0 \qquad (2.3.9)$$

n anındaki katsayı hata vektörü , $\mathbf{w}(n)$ den dolayı bir raslantı vektörüdür. (2.3.9) eşitliğine benzetilerek $\mathbf{e}_w(n+1)$

$$\mathbf{e}_w(n+1) = \mathbf{w}(n+1) - \mathbf{w}_0 \qquad (2.3.10)$$

yazılıp $\mathbf{w}(n+1)$ yerine (2.3.7) deki ifadesi kullanılırsa

$$\mathbf{e}_w(n+1) = [\, \mathbf{I} - \mu.\mathbf{x}(n).\mathbf{x}^T(n) \,].\mathbf{w}(n) + \mu.\mathbf{x}(n).d(n) - \boldsymbol{w}_0$$

$$\mathbf{e}_w(n+1) = [\, \mathbf{I} - \mu.\mathbf{x}(n).\mathbf{x}^T(n) \,].[\, \mathbf{w}(n) - w_0 \,] + \mu[\, \mathbf{x}(n).d(n) - \mathbf{x}(n).\mathbf{x}^T(n)\boldsymbol{w}_0]$$

$$\mathbf{e}_w(n+1) = [\, \mathbf{I} - \mu.\mathbf{x}(n).\mathbf{x}^T(n) \,].\, \mathbf{e}_w(n) + \mu[\, \mathbf{x}(n).d(n) - \mathbf{x}(n).\mathbf{x}^T(n)\boldsymbol{w}_0 \,]$$

(2.3.11)

elde edilir. (2.3.11) eşitliği ile verilen fark denklemlerindeki vektörler birer raslantı vektörüdürler. Bu nedenle kararlılık incelemesine (2.3.11) eşitliğinin her iki tarafının ortalaması alınarak devam edilmesi uygun olur. Bu aşamada x(n) ve d(n) süreçleriyle ilgili olarak aşağıdaki varsayımlar geçerli sayılacaktır [1],[7].

- Giriş vektörleri $\mathbf{x}(1)$, $\mathbf{x}(2)$, ..., $\mathbf{x}(n)$ bağımsız vektörlerdir.

$$E\,[\, \mathbf{x}(n).\mathbf{x}^T(k) \,] = \mathbf{0} \quad , \quad k= 1,2, \ldots , (n-1)$$

- n anındaki giriş vektörü $\mathbf{x}(n)$, istenen işaret d(n) in n anından önceki örneklerinden bağımsızdır.

$$E\,[\, \mathbf{x}(n).d(k) \,] = \mathbf{0} \quad , \quad k = 1,2, \ldots , (n-1)$$

- n anındaki istenen işaret d(n) , n anındaki giriş vektörü $\mathbf{x}(n)$ e bağımlı , istenen işaretin n anından önceki örneklerinden bağımsızdır.

$$E\,[\, \mathbf{x}(n).d(n) \,] \neq \boldsymbol{0}$$

$$E\,[d(k).d(n)\,] = 0\,, \quad k = 1,2,\,.\,.\,,(n-1)$$

Yukarıdaki varsayımlar kabul edilerek (2.3.11) eşitliğinin her iki tarafının ortalaması alınırsa :

$$E[\,\mathbf{e}_w(n+1)\,] = E[\,\mathbf{I} - \mu.\mathbf{x}(n).\mathbf{x}^T(n)\,].\,E[\,\mathbf{e}_w(n)\,] \;+$$

$$\mu.(\,E[\,\mathbf{x}(n).d(n)\,] - E[\,\mathbf{x}(n).\mathbf{x}^T(n)\,].\mathbf{w}_0\,)$$

$$E[\,\mathbf{e}_w(n+1)\,] = [\,\mathbf{I} - \mu.\mathbf{R}\,].E[\,\mathbf{e}_w(n)\,] + \mu.(\,\mathbf{p} - \mathbf{R}.\mathbf{w}_0\,) \qquad (2.3.12)$$

sonucuna varılır. (2.1.11) ile verilmiş olan Wiener Hopf eşitliği kullanılarak (2.3.12) eşitliğinde sağdaki ikinci terimin sıfır olacağı görülür. Bu durumda katsayı hata vektörünün ortalaması ile ilgili olan fark denklemi

$$E[\,\mathbf{e}_w(n+1)\,] = [\,\mathbf{I} - \mu.\mathbf{R}\,].E[\,\mathbf{e}_w(n)\,] \qquad (2.3.13)$$

şekline gelir. (2.3.13) ile verilen fark denklemi (2.2.12) deki fark denklemi ile benzer yapıya sahiptir. LMS algoritmasının optimum Wiener çözümüne yakınsaması veya başka bir ifadeyle katsayı hata vektörünün ortalamasının sıfır vektörü olması için (2.2.9) ile verilen koşulun :

$$0 < \mu < 2 / \lambda_{max}$$

sağlanması gerektiği anlaşımaktadır. Bu koşulun sağlanması durumunda (2.3.13) ile verilmiş olan ifadede katsayı hata vektörünün ortalaması sıfır vektörüne yaklaşmaktadır. Başka bir deyişle LMS algoritması kullanılarak optimum çözüm $\mathbf{w}_0$ için yapılan öngörüde $\mathbf{w}(n)$ vektörünün ortalaması $\mathbf{w}_0$ olmaktadır.

Görüldüğü gibi LMS algoritmasının ortalama anlamında yakınsama koşulu ve hızı adım parametresi μ nün değeri ile ilgilidir. Yakınsama hızını arttırmak amacıyla μ nün her iterasyon adımında değiştirildiği [8],[9],[10] veya birden fazla adım parametresinin kullanıldığı uygulamalar vardır [11],[12]. Ancak bu uygulamalarda μ nün her adımda yeni durumuna uyarlanması için yapılan işlemler *madpr* nin büyümesine sebep olmaktadır.

2.3.2 Ardışıl Enküçük Kareler (RLS) Algoritması

Özilişki matrisi R ve çapraz ilişki vektörü p nin belli bir örnek kümesi için ortalamalarının kullanılması sonucu , R^{-1} in her adımda yeni değerine uyarlanarak hesaplanması ardışıl enküçük kareler algoritmasının temelini oluşturur. Örnek sayısı n olan bir küme için R ve p nin ortalamaları :

$$\mathbf{R}_{ort}(n) = (1/ n) \sum_{k=1}^{n} \mathbf{x}(k).\mathbf{x}^{T}(k) \qquad (2.3.14)$$

$$\mathbf{p}_{ort}(n) = (1/ n) \sum_{k=1}^{n} \mathbf{x}(k).d(k) \qquad (2.3.15)$$

olarak elde edilir. Yukarıda tanımlanan örnek ortalamaları kullanılarak Wiener çözümünün n anındaki değeri bulunmak istenirse :

$$\mathbf{w}(n) = \mathbf{R}^{-1}{}_{ort}(n).\ \mathbf{p}_{ort}(n) \qquad (2.3.16)$$

sonucuna varılır.

$$\mathbf{R}(n) = \sum_{k=1}^{n} \mathbf{x}(k).\mathbf{x}^{T}(k) \qquad (2.3.17)$$

$$\mathbf{p}(n) = \sum_{k=1}^{n} \mathbf{x}(k).d(k) \qquad (2.3.18)$$

tanım bağıntıları kullanılarak (2.3.16) eşitliği yeniden düzenlenirse :

$$\mathbf{w}(n) = \mathbf{R}^{-1}\ (n).\ \mathbf{p}(n) \qquad (2.3.19)$$

elde edilir. Giriş vektörü x(n) ve istenen işaret d(n) nin n anına kadar olan örneklerinin kullanılmasıyla w vektörünün n anındaki öngörülmüş değeri (2.3.19) ile bulunabilir. Bir sonraki adımda w yı yeni durumuna uyarlamak amacıyla :

$$\mathbf{R}(n+1) = \mathbf{R}(n) + \mathbf{x}(n+1).\mathbf{x}^{T}(n+1) \qquad (2.3.20)$$

$$\mathbf{p}(n+1) = \mathbf{p}(n) + \mathbf{x}(n+1).d(n+1) \qquad (2.3.21)$$

şeklinde ardışıl bir bağıntı oluşturulabilir. $\mathbf{R}(n+1)$ matrisinin tersini elde etmek için matris tersi leması kullanılır [13],[14],[15].

A ve B boyutları M olan pozitif tanımlı kare matrisler , D boyutu N olan diğer bir pozitif tanımlı kare matris ve C (M×N) boyutlarında bir matris olmak üzere :

$$\mathbf{A} = \mathbf{B}^{-1} + \mathbf{C}.\mathbf{D}^{-1}.\mathbf{C}^{T} \qquad (2.3.22)$$

şeklinde yazılabiliyorsa A nın tersi :

$$\mathbf{A}^{-1} = \mathbf{B} - \mathbf{B}.\mathbf{C}.(\mathbf{D} + \mathbf{C}^{T}.\mathbf{B}.\mathbf{C})^{-1}.\mathbf{C}^{T}.\mathbf{B} \qquad (2.3.23)$$

olarak bulunur. Özilişki matrisinin tersinin ardışıl olarak hesaplanması amacıyla :

$$\mathbf{A} = \mathbf{R}(n) \quad , \quad \mathbf{B}^{-1} = \mathbf{R}(n-1) \quad , \quad \mathbf{C} = \mathbf{x}(n) \quad \text{ve} \quad \mathbf{D} = 1$$

alınıp bu değerler (2.3.23) eşitliğinde yerine yazılarak :

$$\mathbf{R}^{-1}(n) = \mathbf{R}^{-1}(n-1) - \frac{\mathbf{R}^{-1}(n-1)\,\mathbf{x}(n)\,\mathbf{x}^{T}(n)\,\mathbf{R}^{-1}(n-1)}{1 + \mathbf{x}^{T}(n)\,\mathbf{R}^{-1}(n-1)\,\mathbf{x}(n)} \qquad (2.3.24)$$

özilişki matrisinin tersinin ardışıl olarak hesaplanmasını sağlayan (2.3.24) eşitliği elde edilir.

$$\mathbf{k}(n) = \frac{\mathbf{R}^{-1}(n-1)\,\mathbf{x}(n)}{1 + \mathbf{x}^{T}(n)\,\mathbf{R}^{-1}(n-1)\,\mathbf{x}(n)} \qquad (2.3.25)$$

$$\mathbf{R}^{-1}(n) = \mathbf{R}^{-1}(n-1) - \mathbf{k}(n)\,\mathbf{x}^{T}(n)\,\mathbf{R}^{-1}(n-1) \qquad (2.3.26)$$

$$\alpha(n) = d(n) - \mathbf{w}^{T}(n-1)\,\mathbf{x}(n) \qquad (2.3.27)$$

tanımları yapılarak katsayı vektörü w için :

$$\mathbf{w}(n) = \mathbf{w}(n-1) + \mathbf{k}(n)\,\alpha\,(n) \qquad (2.3.28)$$

(2.3.28) eşitliğiyle verilmiş olan ardışıl enküçük kareler (RLS) algoritması elde edilir.

BÖLÜM 3

DOĞRUSAL SİSTEMLERİN ÇÖZÜMÜNDE ARDIŞIL YÖNTEMLER

Bu çalışmadaki temel düşünce , Wiener-Hopf denklemlerini , doğrusal denklem sistemlerinin ardışıl çözümünde kullanılan yöntemlerle çözmektir. En yaygın bilinen ardışıl yöntemler *Jacobi* ve *Gauss-Seidel* yöntemleridir. Bu çalışmada Jacobi ve Gauss-Seidel yöntemleri Wiener-Hopf denklemlerine ilk olarak uygulanmış ve özilişki matrisinin simetrik yapısından yararlanarak , süzgeç katsayılarını ardışıl olarak hesaplayan bir algoritma üretilmiştir. Özilişki matrisi ve çapraz ilişki vektörünün tam olarak bilinmediği durumda , bunların yerine öngörü değerlerinin kullanılmasıyla *yeni* bir *uyarlamalı kontrol algoritması* elde edilmiştir. Bu yeni algoritmanın bilinen diğer algoritmalarla yakınsama hızı ve işlem sayısı karşılaştırmaları yapılmıştır.

Bu bölümde (2.1.11) eşitliği ile verilmiş olan Wiener - Hopf denklemlerinin doğrusal denklem sistemlerinin ardışıl çözümünde kullanılan bazı yöntemlerle çözümü incelenecektir. Ardışıl yöntemler genellikle büyük boyutlu sistemlerin çözümünde tercih edilmektedirler [16],[17],[18],[19]. Uyarlamalı işaret işlemede kullanılan süzgeç uzunluklarının onlar mertebesinde olduğu gözönünde bulundurulursa bu yöntemlerin Wiener - Hopf denklemlerinin çözümünde kullanılmasının gereksiz olduğu düşünülebilir. Ancak uyarlama işlemi süresinde bir iterasyon adımındaki işlem sayısı *madpr* ve algoritmanın optimum çözüme ulaşması için gerekli iterasyon sayısı incelendiğinde bu yöntemlerin bilinen uyarlamalı algoritmalara göre bir üstünlük sağladığı görülür.

3.1 Jacobi Ardışıl Yöntemi

Doğrusal denklem sistemlerinin ardışıl olarak çözümünde kullanılan yöntemlerin en basiti Jacobi algoritmasıdır. Bu algoritmanın çalışma ilkesini anlamak amacıyla köşegen elemanları sıfır olmayan n boyutlu bir doğrusal denklem sistemi gözönüne alınsın.

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{3.1}$$

(3.1) eşitliği ile verilmiş olan denklem sistemi yeniden düzenlenerek , x_i ve b_i sırasıyla **x** ve **b** vektörlerinin i. satır , a_{ij} **A** matrisinin i . satır , j . sütun elemanı olmak üzere :

$$x_i = \left(b_i - \sum_{j=1}^{i-1} a_{ij}\, x_j - \sum_{j=i+1}^{n} a_{ij}\, x_j \right) / a_{ii} \qquad i=1,....,n \tag{3.2}$$

şeklinde yazılabilir. **x**(k) vektörü $\mathbf{x} = \mathbf{A}^{-1}\,\mathbf{b}$ çözümünün k . adımdaki değeriyse bir sonraki adımdaki **x**(k+1) vektörünün i. elemanı :

$$x_i\,(k+1) = \left(b_i - \sum_{j=1}^{i-1} a_{ij}\, x_j(k) - \sum_{j=i+1}^{n} a_{ij}\, x_j(k) \right) / a_{ii} \qquad i=1,....,n \tag{3.3}$$

olarak elde edilir.

$$\mathbf{D} = \mathrm{diag}\,(a_{11} , a_{22} , ... , a_{nn}) \tag{3.4}$$

$$\mathbf{L} = \begin{bmatrix} 0 & & & \\ a_{21} & 0 & & \\ & & \ddots & \\ a_{n1} & a_{n2} & a_{n,n-1} & 0 \end{bmatrix} \qquad (3.5)$$

$$\mathbf{U} = \begin{bmatrix} 0 & a_{21} & & a_{1n} \\ & 0 & & a_{2n} \\ & & \ddots & \\ & & & a_{n-1,n} \\ & & & 0 \end{bmatrix} \qquad (3.6)$$

(3.4) , (3.5) ve (3.6) eşitlikleriyle tanımlanan **L** , **U** ve **D** matrisleri kullanılarak (3.3) eşitliği ile verilmiş olan ardışıl denklemler matris formunda yazılırsa :

$$\mathbf{D}\,\mathbf{x}(k+1) = -(\mathbf{L} + \mathbf{U})\,\mathbf{x}(k) + \mathbf{b} \qquad (3.7)$$

sonucuna varılır. (3.7) eşitliğinden $\mathbf{x}(k+1)$ çözülürse matris formundaki

$$\mathbf{x}(k+1) = -\mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})\,\mathbf{x}(k) + \mathbf{D}^{-1}\,\mathbf{b} \qquad (3.8)$$

Jacobi algoritması elde edilir [16],[17].

3.2 Jacobi Yönteminin Wiener - Hopf Denklemlerine Uygulanması

Wiener - Hopf denklemlerini Jacobi yöntemini kullanarak çözmek amacıyla özilişki matrisi **R** yi (3.4) , (3.5) , (3.6) ile tanımlanmış olan matrislere benzer şekilde üç matrise ayrıştırmak gerekir. Bu ayrıştırma işlemi yapılınca aşağıdaki M boyutlu $\mathbf{R}_U$, $\mathbf{R}_L$ ve $\mathbf{R}_D$ matrisleri elde edilir.

$$\mathbf{R}_D = \text{diag}\,(r_0 , r_0 , \ldots , r_0) \tag{3.9}$$

$$\mathbf{R}_L = \begin{bmatrix} 0 & & & \\ r_1 & 0 & & \\ \vdots & & \ddots & \\ r_{M-1} & r_{M-2} & r_1 & 0 \end{bmatrix} \tag{3.10}$$

$$\mathbf{R}_U = \begin{bmatrix} 0 & r_1 & & r_{M-1} \\ & 0 & & r_{M-2} \\ & & \ddots & \vdots \\ & & & r_1 \\ & & & 0 \end{bmatrix} \tag{3.11}$$

(2.1.11) eşitliği ile verilmiş olan Wiener - Hopf denklemlerinin çözümü :

$$\mathbf{w}_0 = \mathbf{R}^{-1}\,\mathbf{p} \tag{3.12}$$

şeklindedir. Bu çözümü ardışıl olarak elde etmek için (3.8) deki Jacobi algoritması kullanılarak ve **R** matrisi (3.9) , (3.10) , (3.11) deki gibi üç matrise ayrıştırılarak **w** vektörünün (k+1). adımdaki değeri :

$$\mathbf{w}(k+1) = -\mathbf{R}_D^{-1} (\mathbf{R}_L + \mathbf{R}_U) \mathbf{w}(k) + \mathbf{R}_D^{-1} \mathbf{p} \qquad (3.13)$$

olarak bulunur.

3.3 Kararlılık İncelemesi

$\mathbf{R}_D$, $\mathbf{R}_L$ ve $\mathbf{R}_U$ matrisleri cinsinden **R** özilişki matrisi :

$$\mathbf{R} = \mathbf{R}_D + \mathbf{R}_L + \mathbf{R}_U \qquad (3.14)$$

şeklinde yazılıp, (3.13) eşitliği yeniden düzenlenerek :

$$\mathbf{w}(k+1) = [\mathbf{I} - (1/r_0) \mathbf{R}] \mathbf{w}(k) + \mathbf{p} / r_0 \qquad (3.15)$$

elde edilir. (3.15) eşitliği ile verilmiş olan fark denklemleri incelendiğinde 2.2.1 bölümünde en dik düşüm algoritması için verilmiş olan (2.2.2) eşitlikleri ile benzer yapıda olduğu görülür. (3.15) eşitliğindeki fark denklemi adım parametresi $\mu = 1/r_0$ olan en dik düşüm algoritması olarak yorumlanabilir. (3.15) deki algoritmanın kararlı olabilmesi için (2.2.9) eşitsizliğinden yararlanılarak :

$$0 < 1 / r_0 < 2 / \lambda_{max} \qquad (3.16)$$

koşulunun sağlanması gerektiği görülür. (3.16) eşitsizliğindeki λ_{max} , özilişki matrisi **R** nin en büyük özdeğeridir. Wiener-Hopf denklemlerinin çözümü için uygulanan Jacobi algoritması , LMS algoritmasının adım parametresi $\mu = 1/r_0$ olan özel bir halidir. Dolayısıyla yakınsama hızı bakımından LMS algoritmasına göre bir üstünlük sağlamayacağı açıktır. Bu nedenle Jacobi algoritmasının uyarlamalı olarak yeniden düzenlenmesine gerek görülmemiştir.

3.4 Gauss - Seidel Ardışıl Yöntemi

Jacobi algoritmasında **x** vektörünün k. adımdaki değeri hesaplanırken (k-1). adımdaki değeri kullanılmaktadır. (3.3) eşitliği incelendiğinde **x** in i. elemanının k. değeri $x_i(k)$ hesaplanırken indisin $j < i$ olduğu durumlarda da $x_j(k-1)$ değerlerinin kullanıldığı görülür. İndisin $j < i$ olduğu durumlarda $x_j(k)$ değerleri daha önceden hesaplanmış oldukları için $x_i(k)$ nın elde edilmesinde bu değerler kullanılabilir. Örneğin $x_2(k)$ nın hesaplanmasında $x_1(k-1)$ yerine $x_1(k)$ değerinden yararlanılabilir. Böyle bir değişiklik yapılıp (3.3) eşitliği yeniden düzenlenirse :

$$x_i(k+1) = \left(b_i - \sum_{j=1}^{i-1} a_{ij}\, x_j(k+1) - \sum_{j=i+1}^{n} a_{ij}\, x_j(k) \right) / a_{ii} \qquad i=1,\ldots,n \qquad (3.17)$$

(3.17) ile verilmiş olan Gauss - Seidel algoritması elde edilmiş olur. (3.17) eşitliğindeki denklemler matris formunda yazılmak istenirse (3.4) , (3.5) , (3.6) ile tanımlanmış olan **L** , **D** , **U** matrisleri kullanılarak:

$$\mathbf{x}(k+1) = -(\mathbf{D}+\mathbf{L})^{-1}\,\mathbf{U}\,\mathbf{x}(k) + (\mathbf{D}+\mathbf{L})^{-1}\,\mathbf{b} \qquad (3.18)$$

şeklinde matrissel formdaki Gauss - Seidel algoritması elde edilir [16],[17],[18].

3.5 Gauss - Seidel Yönteminin Wiener - Hopf Denklemlerine Uygulanması

(3.18) eşitliği ile verilmiş olan Gauss - Seidel algoritması Wiener - Hopf denklemlerine uygulanırsa (3.9) , (3.10) , (3.11) eşitliklerinde tanımlanmış olan köşegen, üstüçgen , altüçgen özilişki matrisleri kullanılarak ve $\mathbf{R}_U = \mathbf{R}_L^T$ özelliğinden yararlanarak :

$$\mathbf{w}(k+1) = -(\mathbf{R}_L + \mathbf{R}_D)^{-1}\,\mathbf{R}_L^T\,\mathbf{w}(k) + (\mathbf{R}_L + \mathbf{R}_D)^{-1}\,\mathbf{p} \qquad (3.19)$$

sonucuna varılır. Katsayı vektörünün i. elemanını ardışıl olarak hesaplayan ifade elde edilmek istenirse özilişki matrisinin simetrik yapısından yararlanarak:

$$w_i(k+1) = \left(p_i - \sum_{j=1}^{i-1} r_{i-j}\, w_j(k+1) - \sum_{j=i+1}^{M} r_{j-i}\, w_j(k) \right) / r_0 \qquad (3.20)$$

eşitliği bulunur. Görüldüğü gibi matris formundaki hesaplamalar matris tersi işlemlerini gerektirdiği halde katsayı vektöründeki elemanların ayrı ayrı hesaplanması işlemi çok daha kısadır.

3.6 Kararlılık İncelemesi

Gauss - Seidel algoritmasının Wiener - Hopf denklemlerine uygulanması sonucu elde edilmiş olan (3.19) eşitliğindeki matris formundaki fark denkleminin (2.2.2) eşitliği ile verilmiş olan en dik düşüm algoritmasının yapısına benzediği görülür. (2.2.2) bölümünde yapılan kararlılık incelemesinden yararlanarak (3.19) ile verilen fark denklemlerinin yakınsaması için $(\mathbf{R}_L + \mathbf{R}_D)^{-1} \mathbf{R}_L^T$ matrisinin özdeğerlerinin birim daire içinde olması gerektiği anlaşılır. Bu koşul pozitif tanımlı simetrik matrisler için daima sağlanır [20]. Özilişki matrisi de pozitif tanımlı simetrik bir matris olduğuna göre (3.19) daki fark denklemi , bir koşul gerekmeden optimum Wiener çözümüne yakınsayacaktır. Bu durumu görebilmek amacıyla (3.19) daki fark denklemleri çözülürse :

$$\mathbf{w}(k) = [-(\mathbf{R}_L + \mathbf{R}_D)^{-1} \mathbf{R}_L^T]^k \mathbf{w}(0) +$$

$$\left\{ \sum_{n=0}^{k-1} [-(\mathbf{R}_L + \mathbf{R}_D)^{-1} \mathbf{R}_L^T]^n \right\} (\mathbf{R}_L + \mathbf{R}_D)^{-1} \mathbf{p} \qquad (3.21)$$

sonucuna varılır. $[- (\mathbf{R}_L + \mathbf{R}_D)^{-1} \mathbf{R}_L^T]$ matrisinin özdeğerleri birim daire içindeyse (3.21) eşitliğinin sağındaki birinci terim k nın büyük değerleri için sıfıra gidecek ve algoritma başlangıç koşullarından bağımsız olacaktır. Eşitliğin sağındaki ikinci terimdeki toplam ifadesi yine $[- (\mathbf{R}_L + \mathbf{R}_D)^{-1} \mathbf{R}_L^T]$ matrisinin özdeğerlerinin birim daire içinde olduğu varsayımıyla k nın büyük değerleri için sonlu kalacaktır. Algoritmanın optimum Wiener çözümüne yakınsadığını görmek amacıyla (2.2.11) eşitliğiyle tanımlanmış olan katsayı hata vektörü kullanılabilir. Katsayı hata vektörünün sıfır vektör olması algoritmanın çözüme ulaşması anlamındadır. Bölüm (2.2.3) dekine benzer işlemlerle Gauss - Seidel algoritması için katsayı hata vektörüyle ilgili fark denklemi :

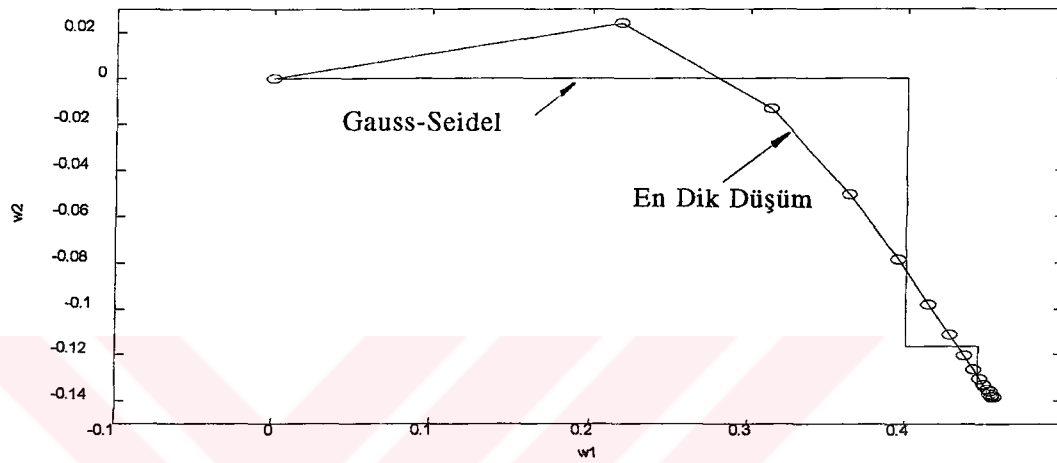
$$\mathbf{e}_w(k+1) = [- (\mathbf{R}_L + \mathbf{R}_D)^{-1} \mathbf{R}_L^T] \mathbf{e}_w(k) \qquad (3.22)$$

olarak bulunur. (3.22) deki fark denkleminin çözümü k nın büyük değerleri için sıfır vektörüne yaklaşacaktır.

3.7 Yakınsama Hızı İncelemesi

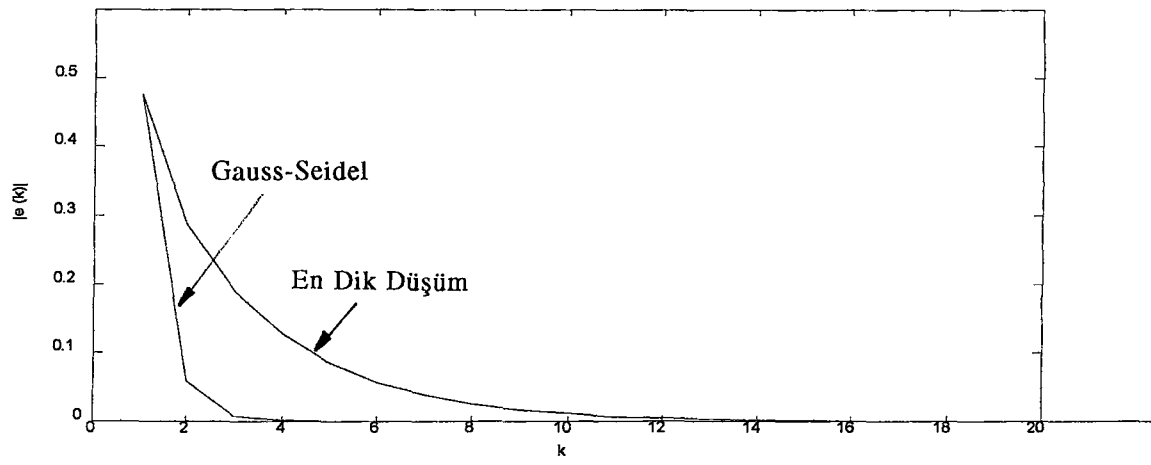
Wiener - Hopf denklemleri için Gauss - Seidel algoritması koşul gerektirmeden yakınsadığı halde , yakınsama hızı özilişki matrisinin yapısından bağımsız değildir. (3.22) ifadesinden anlaşılacağı gibi yakınsama hızı $[(\mathbf{R}_L+\mathbf{R}_D)^{-1} \mathbf{R}_L^T]$ matrisinin oransal özdeğer yaygınlığı ile ilgilidir. Bu çalışmada özilişki matrisi $\mathbf{R}$ nin ve $[-(\mathbf{R}_L + \mathbf{R}_D)^{-1} \mathbf{R}_L^T]$ matrisinin oransal özdeğer yaygınlıkları arasındaki ilişki kapalı bir yapıda verilememiştir. Özilişki matrisinin değişik oransal özdeğer yaygınlığı değerleri için iki boyutlu durumda katsayı vektörünün optimum Wiener çözümüne yaklaşması ve katsayı hata vektörünün normunun sıfıra gitmesi şekil (3.1) - (3.6) da gösterilmiştir. Oransal özdeğer yaygınlığının ve bununla ilgili olan katsayı vektörünün sayısal değerleri [1] numaralı kaynakta kullanılan örneklerden alınmıştır. Aynı özdeğer yaygınlığına sahip özilişki

matrisleri için uygulanan en dik düşüm ve Gauss - Seidel algoritmaları yakınsama hızı açısından karşılaştırıldıklarında Gauss- Seidel algoritmasının yakınsama hızının daha büyük olduğu anlaşılmaktadır.



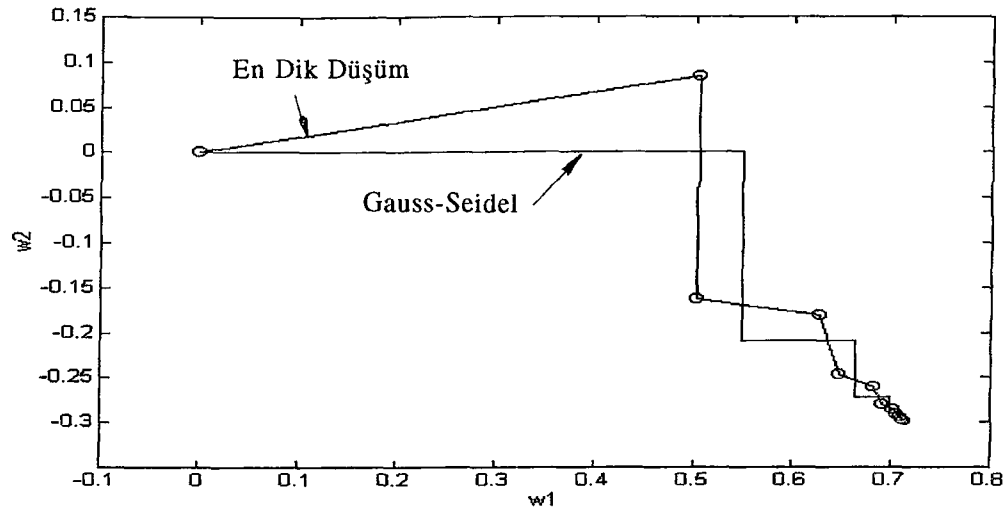
Şekil 3.1

$w_0 = [\ 0.4553 \quad -0.1385\ ]$, $\lambda_{max} / \lambda_{min} = 6$, $\mu = 0.5$



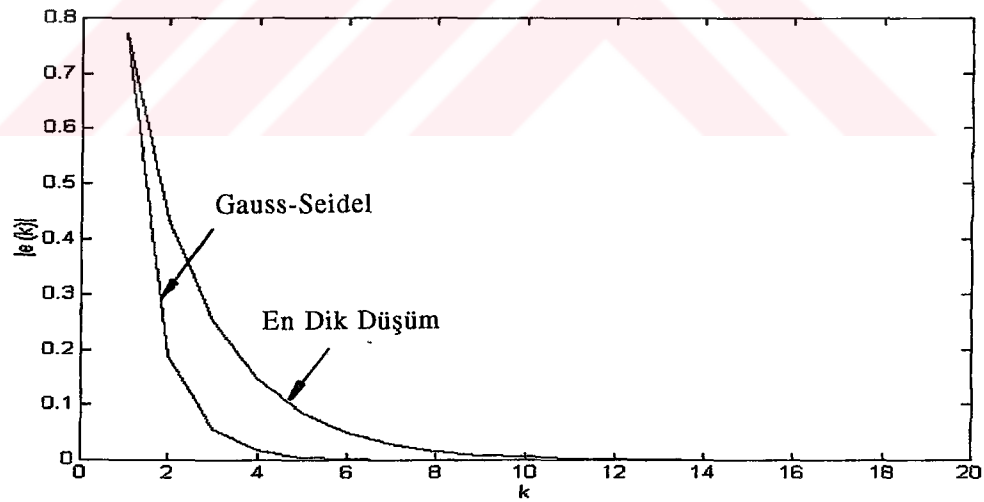
Şekil 3.2

$w_0 = [\ 0.4553 \quad -0.1385\ ]$, $\lambda_{max} / \lambda_{min} = 6$, $\mu = 0.5$



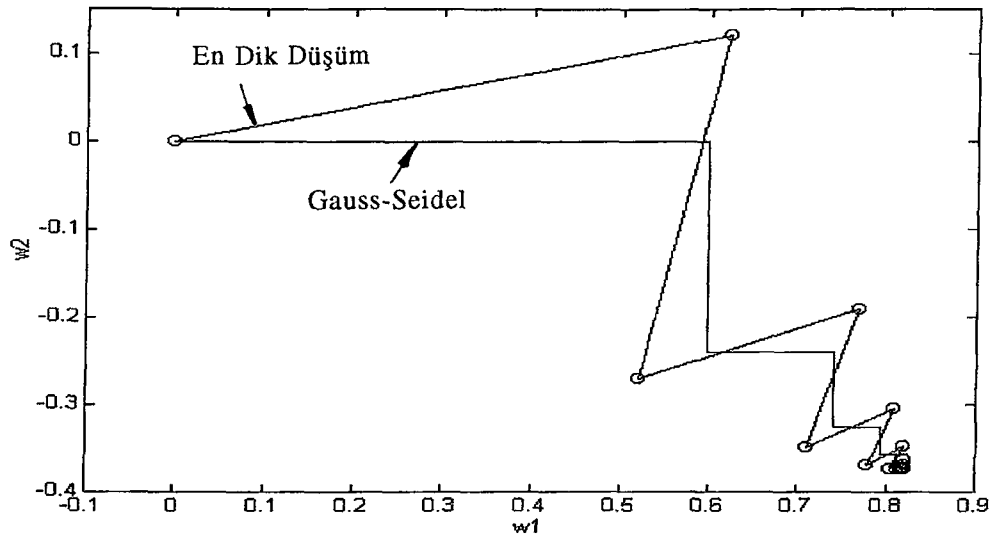
Şekil 3.3

$w_0 = [\ 0.7125 \quad -0.2987\ ]$, $\lambda_{max} / \lambda_{min} = 22$, $\mu = 0.75$



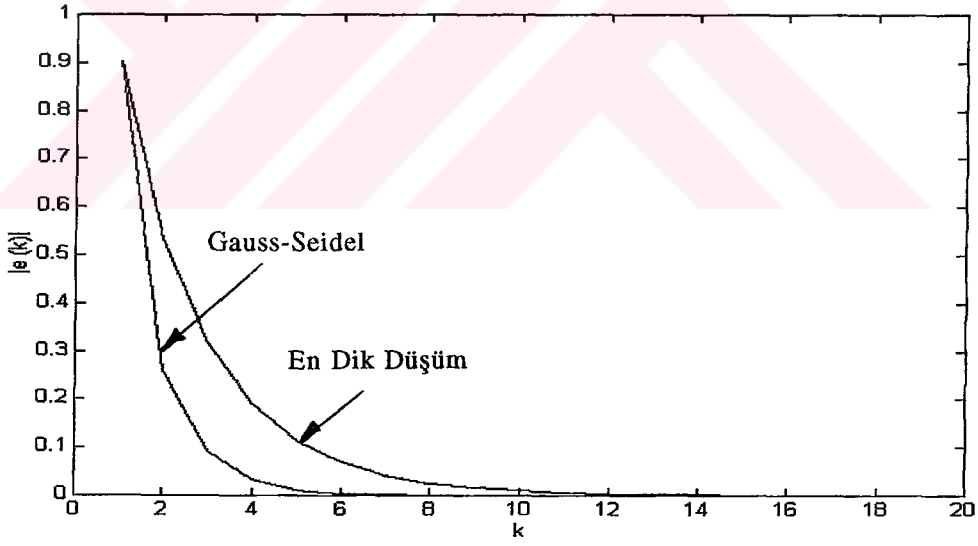
Şekil 3.4

$w_0 = [\ 0.7125 \quad -0.2987\ ]$, $\lambda_{max} / \lambda_{min} = 22$, $\mu = 0.75$



Şekil 3.5

$w_0 = [\ 0.8199 \quad -0.3735\ ]$, $\lambda_{max} / \lambda_{min} = 47$, $\mu = 0.8$



Şekil 3.6

$w_0 = [\ 0.8199 \quad -0.3735\ ]$, $\lambda_{max} / \lambda_{min} = 47$, $\mu = 0.8$

BÖLÜM 4

UYARLAMALI JACOBİ VE GAUSS-SEİDEL ALGORİTMALARI

4.1 Uyarlamalı Jacobi Algoritması

Bu bölümde özilişki matrisi ve çapraz ilişki vektörleri yerine (2.3.14) ve (2.3.15) eşitlikleriyle tanımlanmış örnek ortalamaları kullanılarak Jacobi algoritmasının Wiener - Hopf denklemlerine uygulanması incelenecektir. Örnek sayısı n olan bir küme için özilişki matrisi ve çapraz ilişki vekrörü örnek ortalamları aşağıdaki şekilde tanımlanmıştır.

$$\mathbf{R}_{ort}(n) = 1/n \sum_{k=1}^{n} \mathbf{x}(k).\mathbf{x}^T(k) \qquad (4.1)$$

$$\mathbf{p}_{ort}(n) = 1/n \sum_{k=1}^{n} \mathbf{x}(k).d(k) \qquad (4.2)$$

Özilişki matrisi ve çapraz ilişki vektörlerinin elemanları öngörü değerleri örnek ortalamaları olan özilişki ve çapraz ilişki katsayılarıdır. Bu katsayılar :

$$r_i(n) = 1/n \sum_{k=1}^{n} x(k).x(k-i) \qquad (4.3)$$

$$p_i(n) = 1/n \sum_{k=1}^{n} x(k-i).d(k) \qquad (4.4)$$

eşitlikleriyle tanımlanırlar. (4.3) ile verilmiş olan örnek özilişki katsayıları kullanılarak (3.2) bölümünde (3.9) , (3.10) , (3.11) eşitlikleriyle tanımlanan köşegen , altüçgen ve üstüçgen özilişki matrisleri yeniden yazılırsa :

$$\mathbf{R}_D(n) = \text{diag}\,(r_0(n), r_0(n), \ldots, r_0(n)) \tag{4.5}$$

$$\mathbf{R}_L(n) = \begin{bmatrix} 0 & & & \\ r_1(n) & 0 & & \\ \vdots & & \ddots & \\ r_{M-1}(n) & r_{M-2}(n) & r_1(n) & 0 \end{bmatrix} \tag{4.6}$$

$$\mathbf{R}_U(n) = \begin{bmatrix} 0 & r_1(n) & \cdots & r_{M-1}(n) \\ & 0 & \ddots & r_{M-2}(n) \\ & & \ddots & \vdots \\ & & & r_1(n) \\ & & & 0 \end{bmatrix} \tag{4.7}$$

eşitlikleri elde edilir. Jacobi algoritmasının Wiener - Hopf denklemlerine uygulanması sonucu elde edilmiş olan (3.13) eşitliğindeki fark denklemlerinde köşegen , altüçgen , üstüçgen özilişki matrisleri yerine (4.5) , (4.6) , (4.7) ile tanımlanmış olan örnek ortalamaları yazılarak Uyarlamalı Jacobi algoritması için:

$$\mathbf{w}(n+1) = -\mathbf{R}_D^{-1}(n)\,(\mathbf{R}_L(n) + \mathbf{R}_U(n))\,\mathbf{w}(n) + \mathbf{R}_D^{-1}(n)\,\mathbf{p}(n) \tag{4.8}$$

eşitliğindeki fark denklemi bulunur.

4.2 Uyarlamalı Gauss - Seidel Algoritması

Gauss - Seidel algoritmasının Wiener - Hopf denklemlerine uygulanmasında (4.1) altbölümünde yapıldığı gibi özilişki matrisi ve çapraz ilişki vektörü yerine bunların öngörü değerleri olan örnek ortalamaları kullanılarak (3.19) eşitliği yeniden düzenlenirse :

$$\mathbf{w}(n+1) = - \left(\mathbf{R}_L(n) + \mathbf{R}_D(n) \right)^{-1} \mathbf{R}_L^T(n)\, \mathbf{w}(n) + \left(\mathbf{R}_L(n) + \mathbf{R}_D(n) \right)^{-1} \mathbf{p}(n) \qquad (4.9)$$

eşitliğindeki Uyarlamalı Gauss - Seidel algoritması elde edilir. (3.19) ve (4.9) eşitlikleriyle tanımlanan matris formundaki Gauss - Seidel ve Uyarlamalı Gauss - Seidel algoritmaları kararlılık incelemesi açısından uygun olmalarına rağmen katsayı vektörünün hesaplanmasında işlem kolaylığı ve programlama basitliği sağladığı için (3.20) eşitliği kullanılır. (3.20) eşitliğinde özilişki ve çaprazilişki katsayıları yerine (4.3) ve (4.4) ile tanımlanan örnek öngörü değerleri kullanılarak :

$$w_i(n+1) = \left(p_i(n) - \sum_{j=1}^{i-1} r_{i-j}(n)\, w_j(n+1) - \sum_{j=i+1}^{M} r_{j-i}(n)\, w_j(n) \right) / r_0(n) \qquad (4.10)$$

sonucuna varılır. (4.10) eşitliği incelenirse özilişki ve çapraz ilişki katsayılarının (4.3) ve (4.4) deki tanımları yerine

$$r_i(n) = \sum_{k=1}^{n} x(k).x(k-i) \qquad (4.11)$$

$$p_i(n) = \sum_{k=1}^{n} x(k-i).d(k) \qquad (4.12)$$

eşitlikleriyle verilen öngörü değerlerinin kullanılabileceği analşılır. (4.11) ve (4.12) deki öngörü değerlerinin (n+1) adımındaki durumlarına uyarlanmaları

$$r_i(n+1) = \sum_{k=1}^{n} x(k).x(k-i) + x(n+1)x(n+1-i)$$

$$r_i(n+1) = r_i(n) + x(n+1)x(n+1-i) \qquad (4.13)$$

$$p_i(n+1) = \sum_{k=1}^{n} x(k-i).d(k) + x(n+1-i)d(n+1)$$

$$p_i(n+1) = p_i(n) + x(n+1-i)d(n+1) \qquad (4.14)$$

eşitlikleriyle yapılabilir. İki boyutlu durumda katsayı vektörü elemanlarının ardışıl olarak hesaplanması istenirse :

$$w_1(n+1) = (p_1(n) - r_1(n)w_2(n)) / r_0(n) \qquad (4.15)$$

$$w_2(n+1) = (p_1(n) - r_1(n)w_1(n+1)) / r_0(n) \qquad (4.16)$$

sonucuna varılır. Algoritmanın yazılım olarak gerçeklenmesi durumunda , hesaplama (4.15) , (4.16) sırasıyla yapılırsa , önce w_1 sonra w_2 hesaplanırsa , programlama mantığı gereği n indisinin kullanılmasına gerek yoktur. Bu durumda (4.15) ve (4.16) eşitliklerindeki hesaplamalar birbirini izleyen iki program satırı olarak :

$$w_1 = (p_1 - r_1 w_2) / r_0$$

$$w_2 = (p_1 - r_1 w_1) / r_0$$

şeklinde yapılabilir. Böyle bir programlama tekniği basit olması , daha az bellek kullanması ve bunun sonucu olarak daha az bellek erişimiyle , n indisinin kullanıldığı bir programlamaya göre üstünlük gösterir. LMS ve RLS algoritmaları için bu şekilde bir programlama tekniği kullanılamaz. Uyarlamalı

Gauss - Seidel algoritmasının , LMS ve RLS algoritmalarına göre programlama tekniği bakımından bir üstünlük sağladığı söylenebilir.

4.3 Kararlılık incelemesi

Uyarlamalı Gauss - Seidel algoritmasının kararlılık incelemesinde (2.3.1.1) altbölümünde LMS için yapılmış olan varsayım ve işlemlerin benzerlerinden yararlanılabilir. (4.9) ve (4.10) eşitlikleriyle verilmiş olan fark denklemlerinde özilişki ve çapraz ilişki katsayılarının örnek boyutu n olan bir kümeden hesaplanan öngörü değerleri kullanılmaktadır. Bu öngörü değerleri her örnek kümesi için birer raslantı değişkenidirler. Sonuç olarak bu raslantı değişkenlerini kullanarak ardışıl hesaplanan ağırlık katsayı vektörü ve elemanları da birer raslantı değişkeni olacaktır. Kararlılık incelemesinin raslantı değişkenlerinin ortalamaları için yapılması uygun olur. Örnek özilişki katsayılarının dağılımı ve dağılım parametrelerinin bilinmesi bu incelemede kolaylık sağlayacaktır.

4.3.1 Örnek Özilişki Katsayılarının Dağılımı

Normal dağılım gösteren M boyutlu bir $\mathbf{x}$ raslantı vektörünün olasılık yoğunluk fonksiyonu

$$f(\mathbf{x}) = (1/ (2\pi)^{M/2} \, |\Sigma|^{1/2}) \exp [- 1/2 (\mathbf{x} - \mathbf{m}_x)^T \Sigma^{-1} (\mathbf{x} - \mathbf{m}_x)] \qquad (4.17)$$

eşitliğiyle tanımlanır. (4.17) eşitliğinde Σ ve $\mathbf{m}_x$ aşağıda tanımları verilmiş olan $\mathbf{x}$ vektörüne ait kovaryans matrisi ve ortalama vektörüdür.

$$\mathbf{x} = (x_1 \; x_2 \; x_3 \; \ldots \; x_M)$$

$$\Sigma = E [\mathbf{x} \mathbf{x}^T] \qquad \mathbf{m}_x = E [\mathbf{x}]$$

Σ matrisinin i. satır j. sütun elemenı $r_{ij} = E(x_i x_j)$ nin örnek boyutu n olan bir küme için öngörülmüş değeri :

$$r_{ij}(n) = 1/n \sum_{k=1}^{n} x_{ik} x_{jk} \tag{4.18}$$

eşitliğiyle tanımlanır. Örnek boyutu n olan bir kümeden öngörülen özilişki katsayısının olasılık yoğunluk fonksiyonu :

$$f(r(n)) = (n-2)/\pi (1-r^2)^{(n-1)/2} (1-r(n)^2)^{(n-4)/2} \int_0^1 x^{n-2} / (1-r(n) r x)^{n-1} (1-x^2)dx \tag{4.19}$$

olarak elde edilebilir [21] .(4.19) eşitliğinde r özilişki katsayısının gerçek , r(n) örnek kümeden öngörülen değeridir. Örnek özilişki katsayısının ortalaması ve varyansı (4.19) eşitliğinden hesaplanabilir. Örnek sayısının $n > 500$ olduğu durumlarda örnek özilişki katsayısının ortalaması ve varyansı için aşağıda verilmiş olan yaklaşık değerleri kullanılabilir [21].

$$E[r(n)] \cong r \tag{4.20}$$

$$E[(r(n) - r)^2] \cong (1 - r^2)^2 / n \tag{4.21}$$

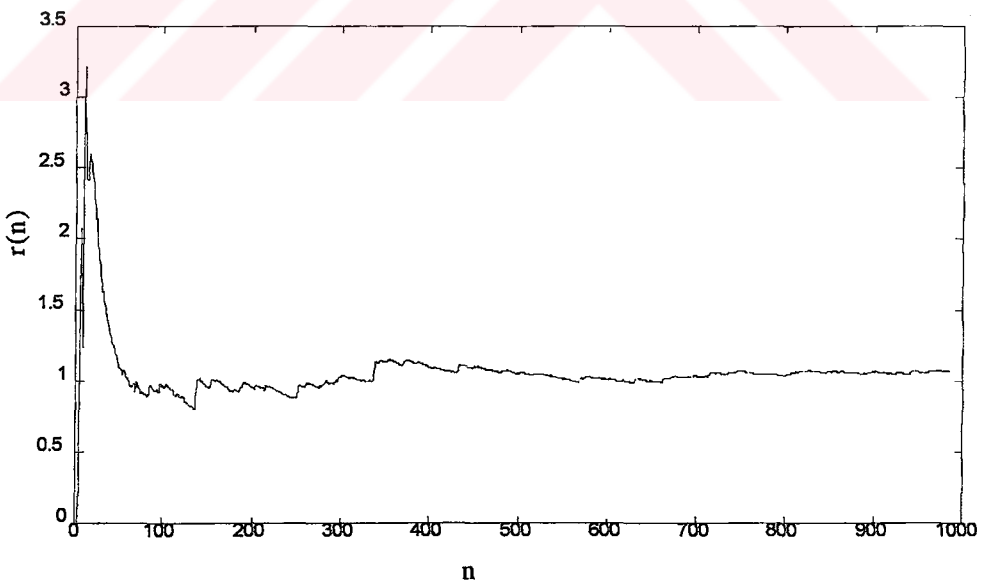
(4.20) ve (4.21) eşitliklerinden anlaşılacağı gibi örnek sayısı büyüdükçe r(n) raslantı değişkeninin varyansı sıfıra yaklaşmaktadır. Bu durumda r(n) raslantı değişkeninin olasılık yoğunluk fonksiyonu da simetrik olmadığı halde impuls fonksiyonuna yaklaşacaktır. Örnek sayısı büyük kümeler için örnek özilişki katsayıları artık bir raslantı değikeni değil de r gibi sabit bir sayı olarak yorumlanabilir.

Örnek özilişki katsayısının değişik örnek değerleri için değişimi şekil (4.1) de verilmiştir. Şekil (4.1) deki örnek özilişki katsayısı r(n) , n değişimi ;

$x(n) = 0.82\, x(n-1) + v(n)$ fark denkleminden ;

$$r(n) = 1/n \sum_{k=1}^{n} x(k)\, x(k-1) \qquad (4.22)$$

eşitliği kullanılarak elde edilmiştir. (4.22) eşitliğinde v(n) sıfır ortalamalı , normal dağılımlı beyaz gürültüdür. Örnek özilişki katsayısının ortalaması yaklaşık olarak n=200 örnekten sonra sabit kalmaktadır. Ortalama civarındaki saçılmanın yaklaşık n=500 örnekten sonra sıfıra yakın bir değerde olduğu söylenebilir.



Şekil 4.1

r(n) , n değişimi

4.3.2 Ortalama Anlamında Yakınsama

Uyarlamalı Gauss - Seidel algoritması için verilmiş olan (4.9) eşitliği yeniden düzenlenerek :

$$(\mathbf{R}_L(n) + \mathbf{R}_D(n)) \mathbf{w}(n+1) = - \mathbf{R}_L^T(n) \mathbf{w}(n) + \mathbf{p}(n) \qquad (4.23)$$

şeklinde yazılabilir. (4.23) eşitliğinde her iki tarafın ortalaması alınarak :

$$E [(\mathbf{R}_L(n) + \mathbf{R}_D(n)) \mathbf{w}(n+1)] = - E [\mathbf{R}_L^T(n) \mathbf{w}(n)] + E [\mathbf{p}(n)] \qquad (4.24)$$

sonucuna varılır. Bu aşamada büyük örnek kümeleri için örnek özilişki matrisinin elemanlarının raslantı değişkeni değil de birer sabit olduğu varsayımıyla (4.24) eşitliği yeniden düzenlenirse :

$$(\mathbf{R}_L + \mathbf{R}_D) E [\mathbf{w}(n+1)] = - \mathbf{R}_L^T E [\mathbf{w}(n)] + \mathbf{p} \qquad (4.25)$$

eşitliği elde edilir.

$$E [\mathbf{w}(n+1)] \cong E [\mathbf{w}(n)]$$

varsayımı kullanılarak (4.25) eşitliği çözülürse :

$$E [\mathbf{w}(n)] = \mathbf{R}^{-1} \mathbf{p}$$

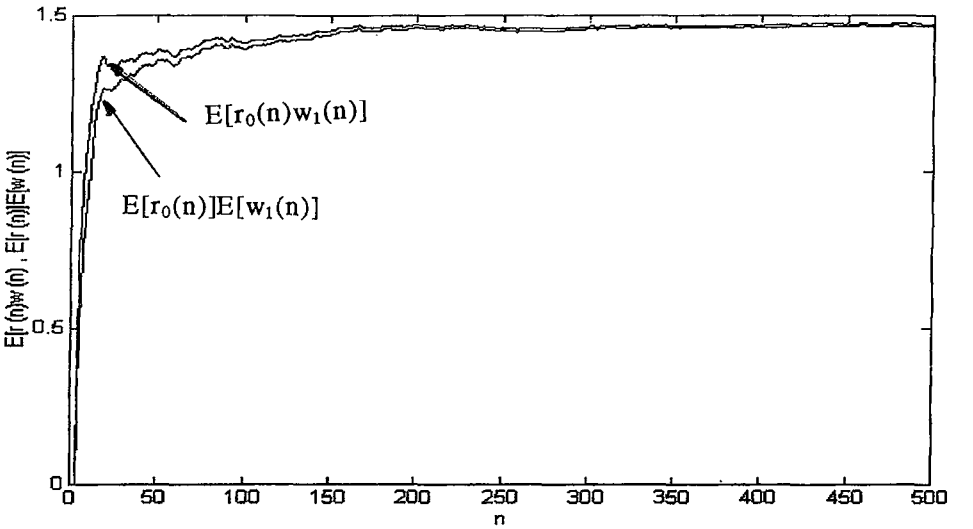
katsayı vektörünün ortalamasının optimum Wiener çözümü olduğu görülür. (4.24) eşitliğinden (4.25) eşitliğine geçişte yapılan varsayım aslında :

$$E\left[\left(\mathbf{R}_L(n) + \mathbf{R}_D(n)\right)\mathbf{w}(n+1)\right] = E\left[\left(\mathbf{R}_L(n) + \mathbf{R}_D(n)\right)\right] E\left[\mathbf{w}(n+1)\right] \quad (4.26)$$

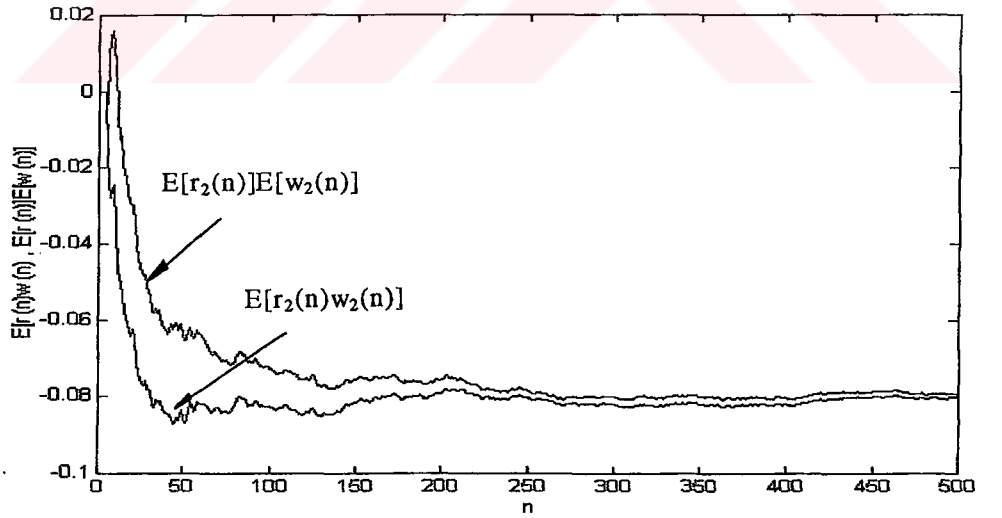
ifadesinin kabulüne dayanmaktadır. (4.26) eşitliğindeki varsayım ilgilenilen her özilişki ve ağırlık katsayısının birbiriyle ilişkisiz olması anlamına gelmektedir. Bu varsayımın yaklaşık olarak sağlandığı sayısal bir örnek üzerinde incelenebilir. Fark denklemi

$$x(n) = 0.82\, x(n-1) - 0.37\, x(n-2) + v(n)$$

şeklinde verilen bir x(n) sürecinden elde edilmiş olan örnekler kullanılarak özilişki katsayıları ve Gauss - Seidel algoritması ile hesaplanan ağırlık katsayılarının ortalamalarının çarpımı ve çarpımlarının ortalaması şekil (4.2) ve şekil (4.3) de gösterilmiştir. Fark denklemindeki v(n) işareti normal dağılımlı beyaz gürültüdür. Özilişki ve ağırlık katsayılarının ortalamaları bağımsız 200 deneme üzerinden hesaplanmıştır. İki boyutlu özilişki matrisi ve ağırlık vektörünün bütün elemanlarının birbirleriyle ilişkisiz oldukları anlaşılmaktadır.



Şekil 4.2 $E[r_0(n)w_1(n)]$, n ve $E[r_0(n)]E[w_1(n)]$, n değişimi



Şekil 4.3 $E[r_2(n)w_2(n)]$, n ve $E[r_2(n)]E[w_2(n)]$, n değişimi

Örnek boyutu 500 olan , 200 bağımsız deneme üzerinden hesaplanmış özilişki katsayılarının ortalamaları $E[r_0(n)] \cong 1.8$, $E[r_1(n)] \cong 1.05$ ve $E[r_2(n)] \cong 0.225$ olarak elde edilmiştir. Uyarlamalı Gauss - Seidel algoritması kullanılarak öngörülmüş olan $w_1(n)$ ve $w_2(n)$ ağırlık katsayılarının ortalamaları $E[w_1(n)] \cong 0.82$ ve $E[w_2(n)] \cong -0.37$ bulunmuştur. $E[r_0(n)]\,E[w_1(n)]$ ve $E[r_0(n)w_1(n)]$, $E[r_2(n)]\,E[w_2(n)]$ ve $E[r_2(n)w_2(n)]$ ortalamalarının zamanla değişimleri sırasıyla şekil (4.2) , şekil (4.3) de verilmiştir. Benzer şekilde her $r_i(n)$ ve $w_j(n)$ için :

$$E[r_i(n)]\,E[w_j(n)] \cong E[r_i(n)w_j(n)]$$

yaklaşık eşitliğinin sağlandığı görülebilir. Her bir özilişki katsayısı ile ağırlık katsayısının ortalamalarının çarpımlarının , çarpımlarının ortalamasına eşit olması (4.26) eşitliğiyle verilen :

$$E\left[\left(\mathbf{R}_L(n) + \mathbf{R}_D(n)\right)\mathbf{w}(n+1)\right] = E\left[\left(\mathbf{R}_L(n) + \mathbf{R}_D(n)\right)\right] E\left[\mathbf{w}(n+1)\right]$$

ifadesinin geçerli olduğunu göstermektedir.

4.4 Durağan Olmayan Ortamlarda Uyarlamalı Gauss Seidel Algoritması

Durağanlık için , dar anlamda (strictly stationary) ve geniş anlamda (wide sense stationary) durağan olmak üzere iki tanım verilmiştir [1]. Bir sürecin istatistiksel özellikleri zamanla değişmiyorsa süreç dar veya kısıtlı anlamda durağandır.x(n) , x(n-1).... , x(n-M) şeklinde ayrık zamanda (M+1) boyutlu vektör olarak temsil edilen bir stokastik sürecin ortak olasılık yoğunluk fonksiyonu sabit bir M değeri için , zamanla değişmiyorsa bu süreç dar anlamda durağandır. Pratikte x(n) sürecinden gelen örnekler kullanılarak ortak olasılık yoğunluk fonksiyonu tam olarak belirlenemediğinden sürecin durağanlığı için birinci ve ikinci dereceden momentleri incelenerek bir tanım yapılabilir. Ayrık

zaman serisi olarak temsil edilen stokastik sürecin ortalama değer , özilişki ve kovaryans fonksiyonları

$$m(n) = E[\, x(n) \,] \tag{4.27}$$

$$r\,(n,n-k) = E[\, x(n)x(n-k) \,] \tag{4.28}$$

$$c\,(n,n-k) = E[\, (\, x(n) - m(n) \,)(\, x(n-k) - m(n-k) \,) \,] \tag{4.29}$$

eşitlikleriyle tanımlıdır. Stokastik sürecin istatistiksel özellikleri zamanla değişmiyorsa (4.27) , (4.28) ve (4.29) eşitlikleri

$$m(n) = m \tag{4.30}$$

$$r\,(n,n-k) = r(k) \tag{4.31}$$

$$c\,(n,n-k) = c\,(k) \tag{4.32}$$

şeklinde yazılabilir. (4.30) , (4.31) ve (4.32) eşitliklerinin sağlanması sürecin dar anlamda durağan olması için yeterli değildir. Ancak bu eşitlikler sağlanıyorsa süreç geniş anlamda durağan veya ikinci dereceden durağandır denir. Pratik olarak yapılan ölçmeler kullanılarak incelenebilmesi ve süreç üzerinde yapılan doğrusal işlemlere elverişli olması bakımından geniş anlamda durağanlık tanımı dar anlamda durağanlık tanımına göre daha uygundur.

Uyarlamalı Gauss - Seidel ve RLS algoritmalarında örnek özilişki katsayıları kullanılmaktadır. Eğer süreç durağan değilse özilişki katsayılarının öngörü değerlerinde de zamana göre değişim söz konusudur. Algoritmalar kullanılarak öngörülen katsayı vektörü sürecin son durumuyla ilgili olmalıdır. Bunun sağlanması için özilişki katsayıları öngörülürken kullanılan , sürece ait örneklerin zaman domeninde en son gelenlerinin etkisinin arttırılması daha önceki örneklerin ise etkisinin azaltılması gerekir. Bu amaçla

$$\beta(n,k) = \lambda^{n-k} \qquad 0 < \lambda < 1 \tag{4.33}$$

şeklinde tanımlanan bir ağırlaştırma (weighting factor , forgetting factor) katsayısından yararlanılır. (2.3.17) ve (2.3.18) eşitlikleriyle verilen örnek özilişki matrisi ve çapraz ilişki vektörleri (4.33) deki ağırlaştırma katsayısı kullanılarak yeniden yazılırsa

$$\mathbf{R}(n) = \sum_{k=1}^{n} \lambda^{n-k}\, \mathbf{x}(k)\mathbf{x}^{T}(k) \tag{4.34}$$

$$\mathbf{p}(n) = \sum_{k=1}^{n} \lambda^{n-k}\, \mathbf{x}(k)d(k) \tag{4.35}$$

elde edilir [22]. (4.34) ve (4.35) eşitlikleri incelenirse , k değeri büyüdükçe yani örnek sayısı n olan bir kümede zaman domeninde son gelmiş olan örneklere yaklaştıkça bu örneklerin ağırlaştırma katsayısının büyüdüğü görülür. İlk gelmiş olan örnekler için yani k nın küçük olduğu durumlarda ağırlaştırma katsayısının da küçüleceği anlaşılır. Böylece sürecin son durumuna ait örneklerin etkisi artmakta , daha önceki örneklerin etkisi azalmaktadır. (4.34) ve (4.35) eşitliklerinde k = n ile ilgili olan son terimler toplamın dışında bırakılarak eşitlikle yeniden düzenlenirse

$$\mathbf{R}(n) = \lambda \left(\sum_{k=1}^{n-1} \lambda^{n-1-k}\, \mathbf{x}(k)\mathbf{x}^{T}(k)\right) + \mathbf{x}(n)\mathbf{x}^{T}(n) \tag{4.36}$$

$$\mathbf{p}(n) = \lambda \left(\sum_{k=1}^{n-1} \lambda^{n-1-k}\, \mathbf{x}(k)d(k)\right) + \mathbf{x}(n)d(n) \tag{4.37}$$

elde edilir. (4.36) ve (4.37) eşitlikleri incelenirse R(n) ve p(n) için ardışıl yapıdaki

$$\mathbf{R}(n) = \lambda\mathbf{R}(n-1) + \mathbf{x}(n)\mathbf{x}^{T}(n) \tag{4.38}$$

$$\mathbf{p}(n) = \lambda\, \mathbf{p}(n-1) + \mathbf{x}(n)d(n) \tag{4.39}$$

bağıntılarının geçerli olduğu görülür. Uyarlamalı Gauss - Seidel algoritmasında görünen diagonal , alt üçgen ve üst üçgen özilişki matrisleri için de (4.34) ve (4.35) eşitliklerindeki gibi bir ağırlaştırma katsayısı kullanılarak ardışıl bir yapı elde edilebilir. Bu durumda $R_D(n)$, $R_L(n)$ ve $R_L^T(n)$ için

$$\mathbf{R}_D(n) + \mathbf{R}_L(n) + \mathbf{R}_L^T(n) = \lambda\,[\,\mathbf{R}_D(n-1) + \mathbf{R}_L(n-1) + \mathbf{R}_L^T(n-1)] + \mathbf{x}(n)\,\mathbf{x}^T(n) \tag{4.40}$$

eşitliği yazılabilir. Özilişki matrisini oluşturan özilişki katsayıları da ağırlaştırma katsayısı kullanılarak

$$r_i(n) = \sum_{k=1}^{n} \lambda^{n-k}\, x(k)x(k-i)$$

$$r_i(n) = \lambda \sum_{k=1}^{n-1} \lambda^{n-k-1}\, x(k)\, x(k-i) + x(n)x(n-i)$$

$$r_i(n) = \lambda\, r_i(n-1) + x(n)x(n-i) \tag{4.41}$$

şeklinde ardışıl olarak elde edilebilir. Benzer şekilde çapraz ilişki katsayıları da ağırlaştırma işlemi uygulanarak

$$p_i(n) = \lambda\, p_i(n-1) + d(n)x(n-i) \tag{4.42}$$

eşitliğindeki gibi ardışıl hesaplanabilir.

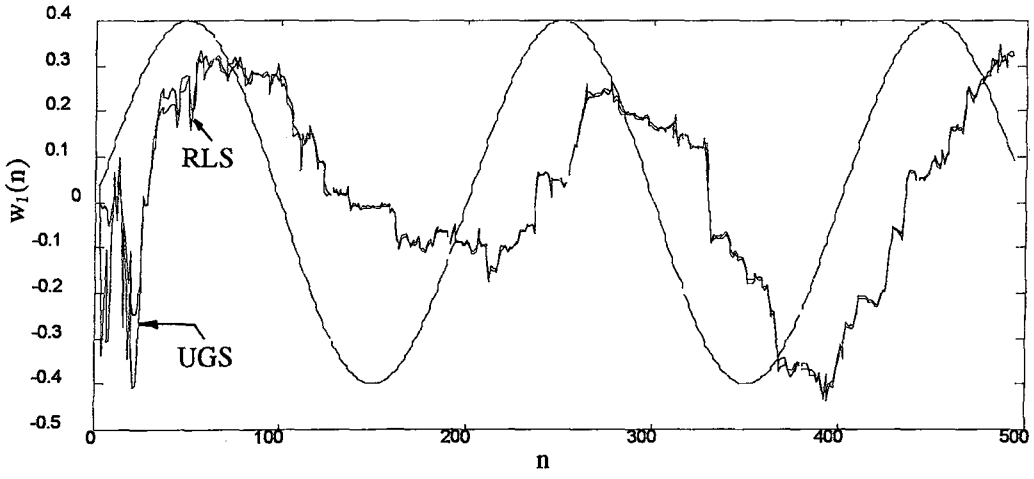
Uyarlamalı Gauss -Seidel algoritmasını durağan olmayan ortamlarda işletmek için yapılması gereken özilişki ve çapraz ilişki katsayılarının , ağırlaştırma etkisi bulunan (4.41) ve (4.42) eşitlikleriyle hesaplanan değerlerinin

kullanılmasıdır. (4.10) eşitliğiyle tanımlanan Uyarlamalı Gauss - Seidel algoritmasında özilişki ve çapraz ilişki katsayıları yerine (4.41) ve (4.42) deki değerleri kullanılarak başka bir değişiklik yapılmaksızın algoritmanın durağan olmayan ortamlar için uygun bir yapıya sahip olması sağlanır.

Durağan olmayan ortamlarda RLS ve Uyarlamalı Gauss-Seidel algoritmaları kullanılarak elde edilen katsayı vektörü kestirimleri karşılaştırmalı olarak şekil(4.4) - şekil(4.19) da verilmiştir. Durağan olmayan ortamı elde etmek için katsayıları zamanla değişen bir AR süreçten yararlanılmıştır.

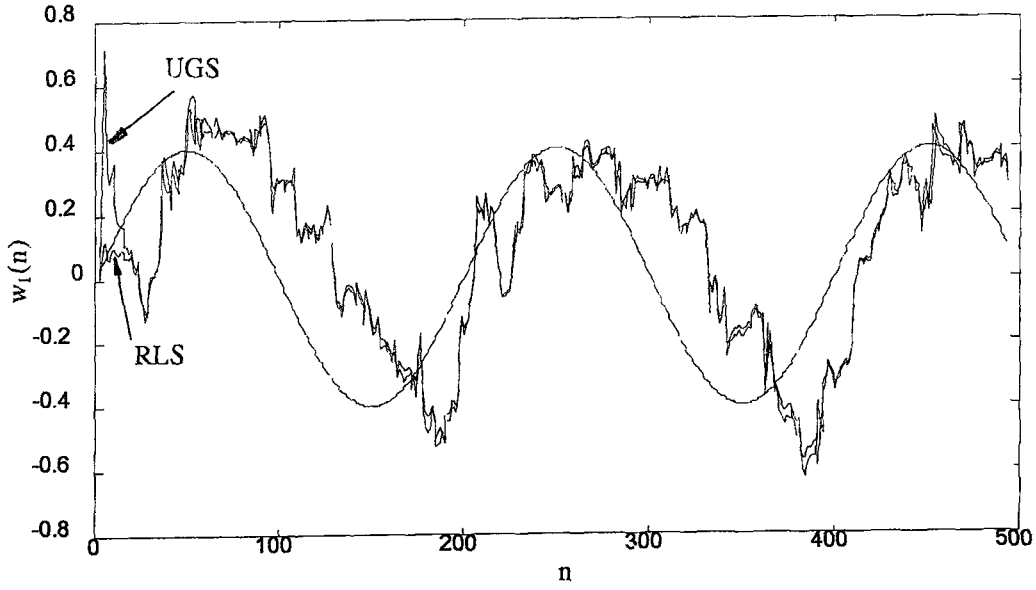
$$x(n) = w_1(n)x(n-1) + w_2(n)x(n-2) + v(n) \qquad (4.43)$$

(4.43) eşitliğiyle verilmiş olan x(n) süreci , w(n) katsayıları zamanla değiştiği için , durağan olmayan bir süreçtir. v(n) süreci varyansı 0.01 olan beyaz gürültüdür. Önce her iki katsayının birden zamanla değiştiği durum incelenmiş , daha sonra katsayılardan birinin sabit diğerinin ise belli bir değerin üzerinde küçük salınımlar yaptığı durum göz önünde bulundurulmuştur. Her iki durum içinde RLS ve Uyarlamalı Gauss-Seidel algoritmaları kullanılarak katsayı vektörünün kestirimi elde edilmiştir.



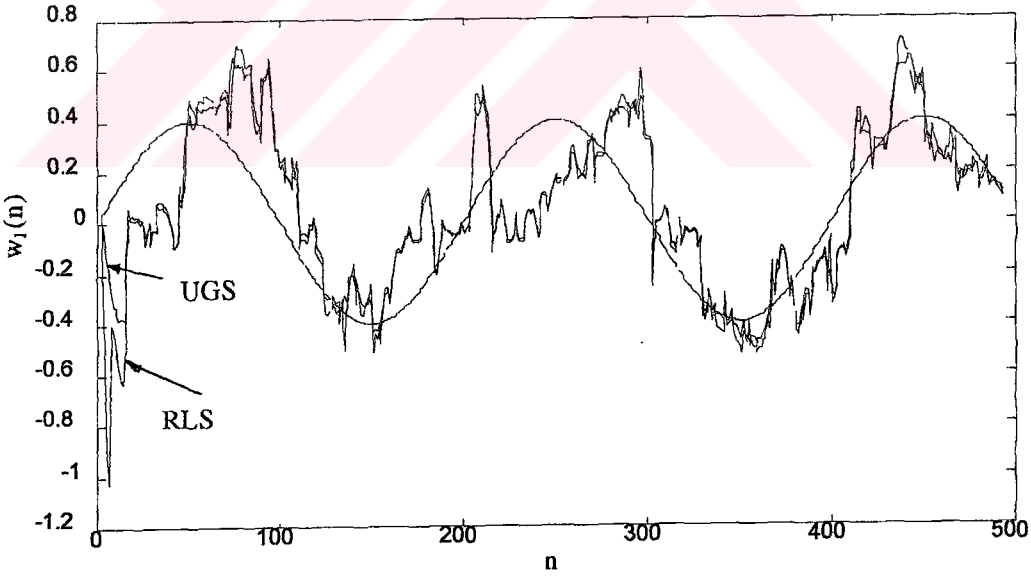
Şekil 4.4

$w_1(n) = 0.4 \sin(2\pi\ 0.005n)$, $w_2(n) = 0.4 \sin(2\pi\ 0.005n)$, $\lambda_R = 0.98$, $\lambda_G = 0.98$



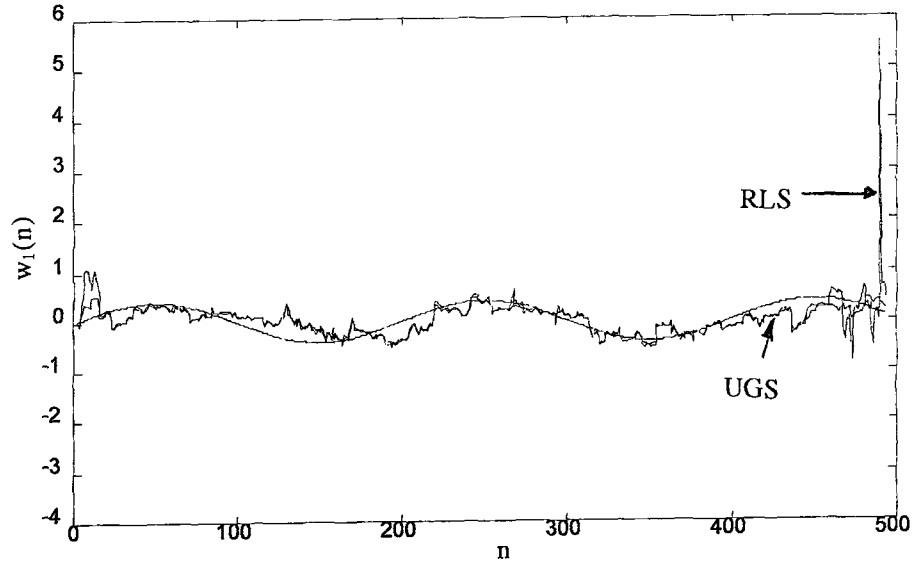
Şekil 4.5

$w_1(n) = 0.4 \sin(2\pi\ 0.005n)$, $w_2(n) = 0.4 \sin(2\pi\ 0.005n)$, $\lambda_R = 0.95$, $\lambda_G = 0.95$



Şekil 4.6

$w_1(n) = 0.4 \sin(2\pi\ 0.005n)$, $w_2(n) = 0.4 \sin(2\pi\ 0.005n)$, $\lambda_R = 0.93$, $\lambda_G = 0.93$



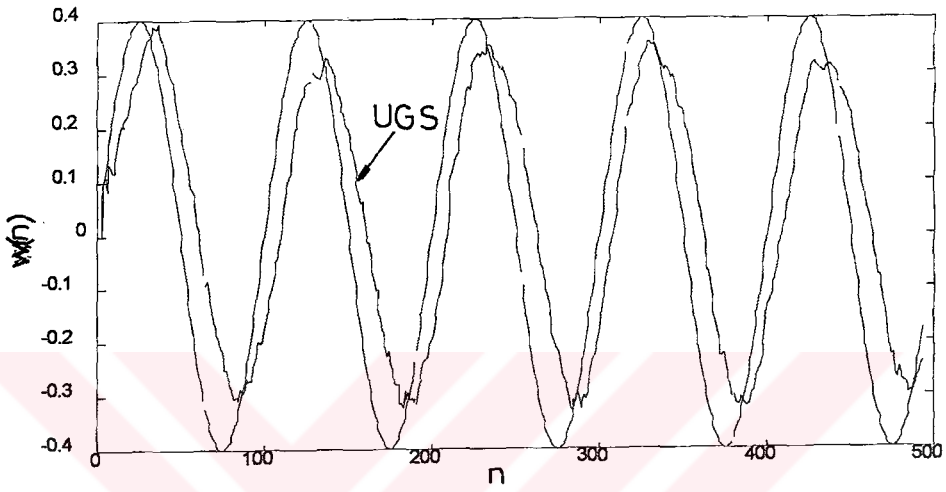
Şekil 4.7

$w_1(n) = 0.4 \sin(2\pi\ 0.005n)$, $w_2(n) = 0.4 \sin(2\pi\ 0.005n)$, $\lambda_R = 0.92$, $\lambda_G = 0.92$

Her iki katsayının da büyük genlikli sinüzoidal değişim gösterdiği durumda , ağırlaştırma faktörünün değişik değerleri için RLS ve Uyarlamalı Gauss-Seidel algoritmaları kullanılarak kestirimi yapılan katsayıların değişimleri şekil.(4.4) - şekil.(4.7) de verilmiştir. Ağırlaştırma faktörleri λ_R ve λ_G nin aynı değerleri için RLS ve Uyarlamalı Gauss-Seidel algoritmaları kullanılarak elde edilen kestirimlerin gerçek katsayı değerlerini izlemesinin aynı kaldığı görülmüştür. Ağırlaştırma katsayısı küçüldükçe izleme performansının arttığı ancak raslantı değişkeni olan katsayı kestireçlerinin varyanslarının büyüdüğü anlaşılmaktadır. RLS algoritması için ağırlaştırma faktörü λ_R nin $\lambda_R < 0.92$ olduğu durumlarda algoritmanın yakınsamayacağı deneysel sonuçlar kullanılarak anlaşılmştır. Literatürde de RLS algoritması için ağırlaştırma faktörünün uygun aralığının $0.9 < \lambda_R < 1$ olduğu söylenmektedir [1].

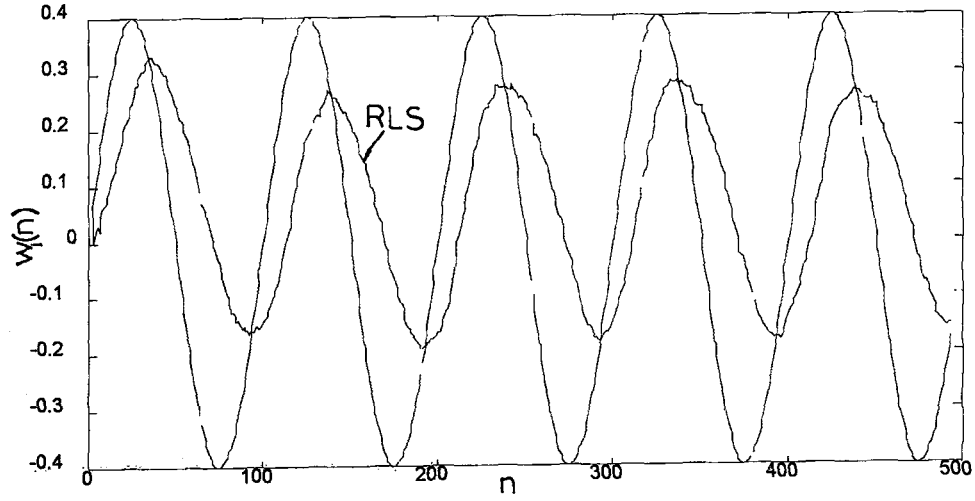
Katsayıların değişiminde frekans arttırılarak RLS ve Uyarlamalı Gauss-Seidel algoritmaları için izleme performansı karşılaştırmaları elde edilmiştir. 100 bağımsız deneme üzerinden alınan kestirim ortalamaları değişimleri , Uyarlamalı Gauss-Seidel algoritması için şekil.(4.8) , RLS algoritması için

şekil.(4.9) da gösterilmiştir. Bu uygulama için RLS algoritmasının ağrlaştırma faktörü $\lambda_R < 0.95$ sınırından küçük değerlerinde yakınsamadığı görülmüştür. Uyarlamalı Gauss-Seidel algoritması için uygun bir ağılaştırma faktörü değeri seçilerek , izleme performansı iyileştirilmiştir.



Şekil 4.8

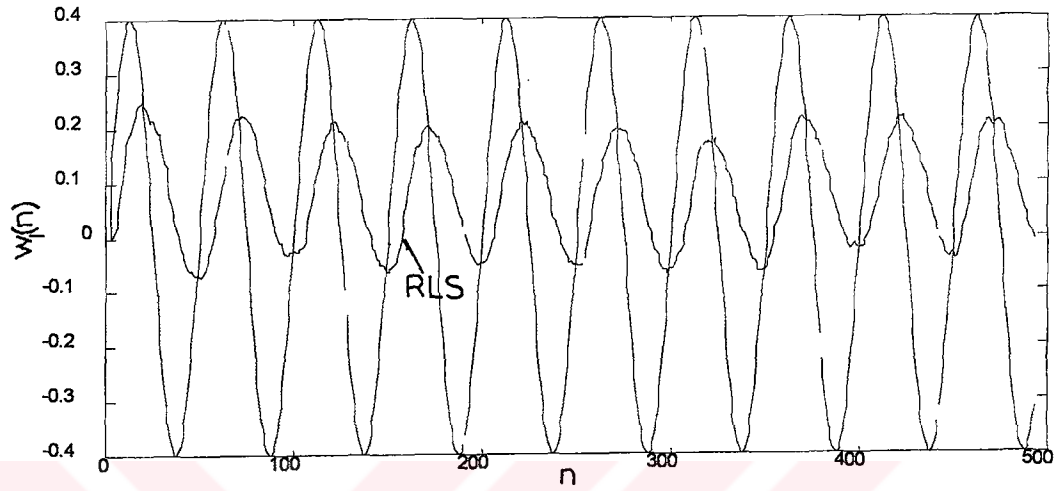
$w_1(n) = 0.4 \sin(2\pi\ 0.01n)$, $w_2(n) = 0.4 \sin(2\pi\ 0.01n)$, $\lambda_G = 0.90$



Şekil 4.9

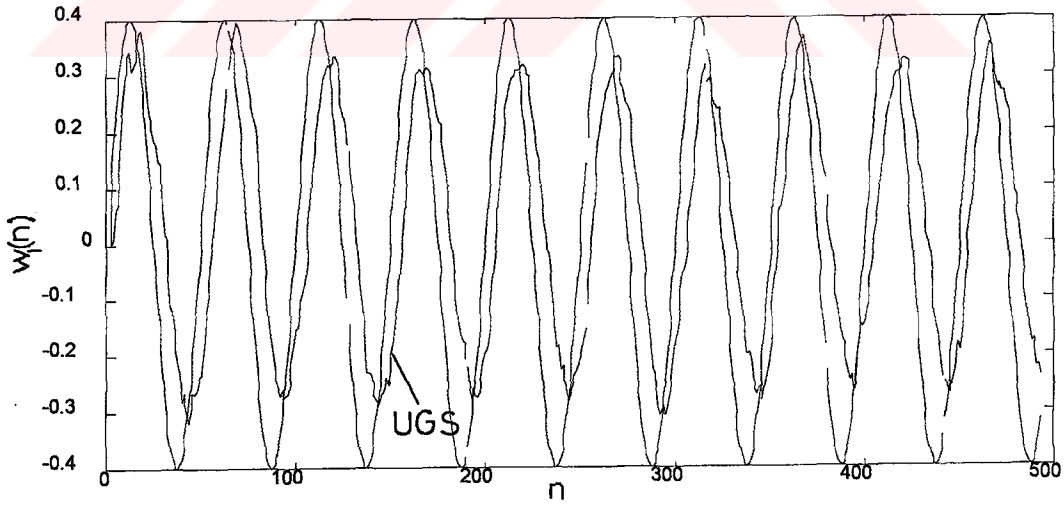
$w_1(n) = 0.4 \sin(2\pi\ 0.01n)$, $w_2(n) = 0.4 \sin(2\pi\ 0.01n)$, $\lambda_R = 0.95$

Şekil.(4.8) ve şekil.(4.9) incelendiğinde Uyarlamalı Gauss-Seidel algoritmasının izleme performansının , RLS algoritmasına göre daha iyi olduğu anlaşılmaktadır.



Şekil 4.10

$w_1(n) = 0.4 \sin(2\pi\ 0.02n)$, $w_2(n) = 0.4 \sin(2\pi\ 0.02n)$, $\lambda_R = 0.95$



Şekil 4.11

$w_1(n) = 0.4 \sin(2\pi\ 0.02n)$, $w_2(n) = 0.4 \sin(2\pi\ 0.02n)$, $\lambda_G = 0.85$

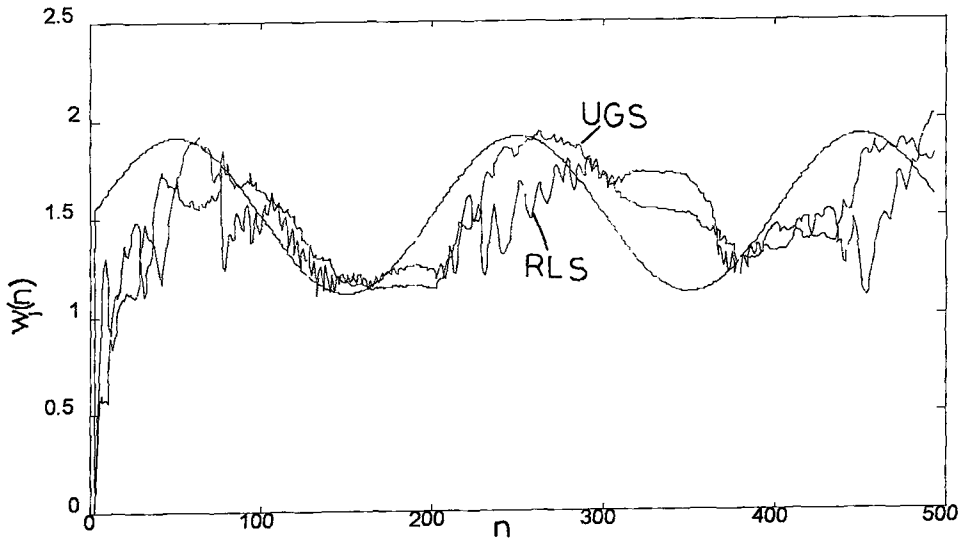
Şekil.(4.10) da RLS , şekil.(4.11) de Uyarlamalı Gauss-Seidel algoritmalarının frekansın bir önceki duruma göre iki katına çıkarıldığı durum için kestirim ortalamalarının değişimleri verilmiştir. Uyarlamalı Gauss-Seidel algoritması için uygun bir ağırlaştırma faktörü ile izleme performansının RLS algoritmasına göre daha iyi olması sağlanabilmektedir. Frekans arttıkça RLS algoritmasının izleme performansının kötüleştiği şekil.(4.9) ve şekil.(4.10) üzerinden görülebilir.

Durağan olmayan ortamı simule etmek için kullanılan ikinci dereceden AR modelde katsayıların her ikisinin birden büyük genlikli değişimleri incelenmiştir. Ancak pratikteki sistemlerde , sistem parametrelerinin büyük genlikli değişimleri yerine sabit değer civarındaki küçük genlikli salınımları daha çok karşılaşılan bir durumdur. Böyle bir durum modelini oluşturmak amacıyla yine ikinci dereceden bir AR modelden yararlanılabilir. (4.43) eşitliğinde

$$w_1(n) = -0.95 \, , \, w_2(n) = 1.514 + 0.4 \sin(2\pi f n)$$

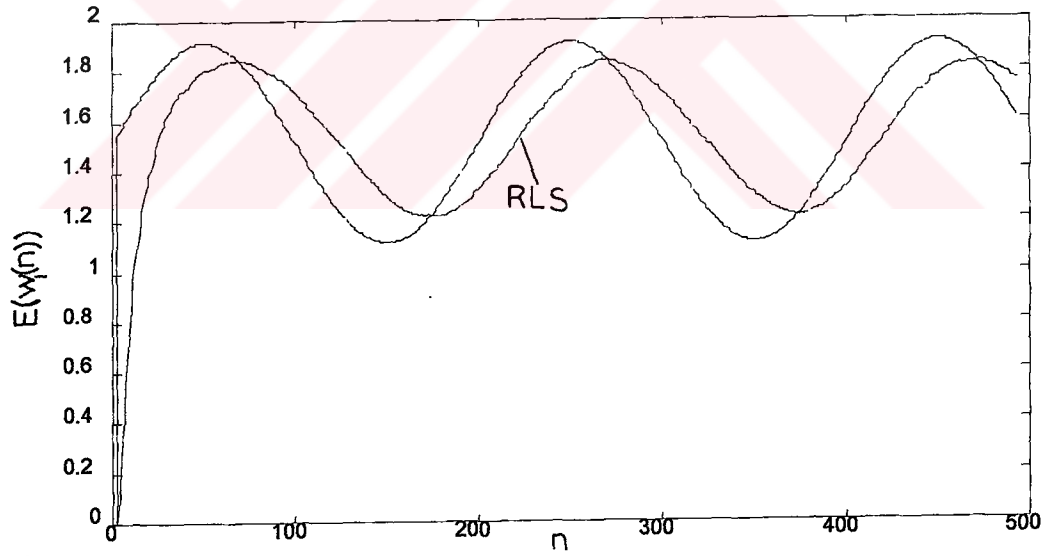
şeklinde bir değişimle istenen durum elde edilebilir. Bu durumda oransal özdeğer yaygınlığının $X(R) = 100$ gibi oldukça büyük bir değeriyle de karşılaşılmaktadır. Yukarıdaki örnek , yakınsama hızının oransal özdeğer yaygınlığına bağımlı olduğu Uyarlamalı Gauss-Seidel algoritmasının durağan olmayan ortamlarda izleme performansının incelenmesi bakımından uygundur.

Ağırlaştırma katsayılarının aynı olduğu durumda RLS ve Uyarlamalı Gauss-Seidel algoritmaları şekil.(4.12) de görüldüğü gibi farklı sonuçlar vermektedir. Bunun sebebi , durağan olmayan ortamda zamanla değişen oransal özdeğer yaygınlığının Uyarlamalı Gauss-Seidel algoritmasının yakınsama hızına olumsuz etkisidir. Katsayıların değişim hızının arttırıldığı bundan sonraki uygulamalarda katsayı kestirimlerinin 100 bağımsız deneme üzerinden hesaplanmış olan ortalamalarının zamanla değişimleri incelenmiştir.



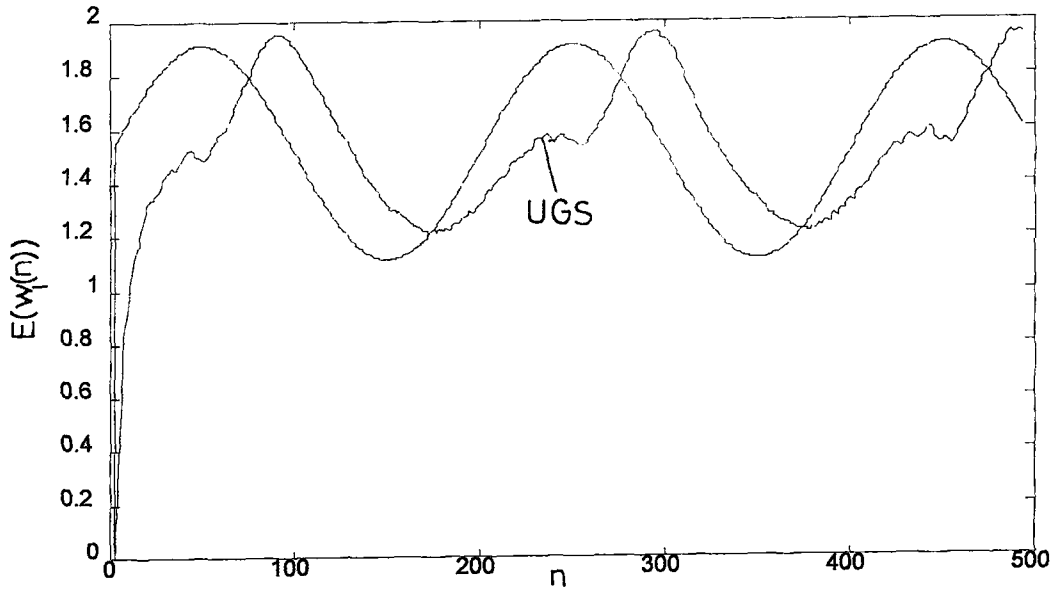
Şekil 4.12

$w_1(n) = 1.5114 + 0.4\ \mathrm{Sin}(2\pi\ 0.005\ n)$, $w_2(n) = -0.95$, $\lambda_R = 0.95$, $\lambda_G = 0.95$



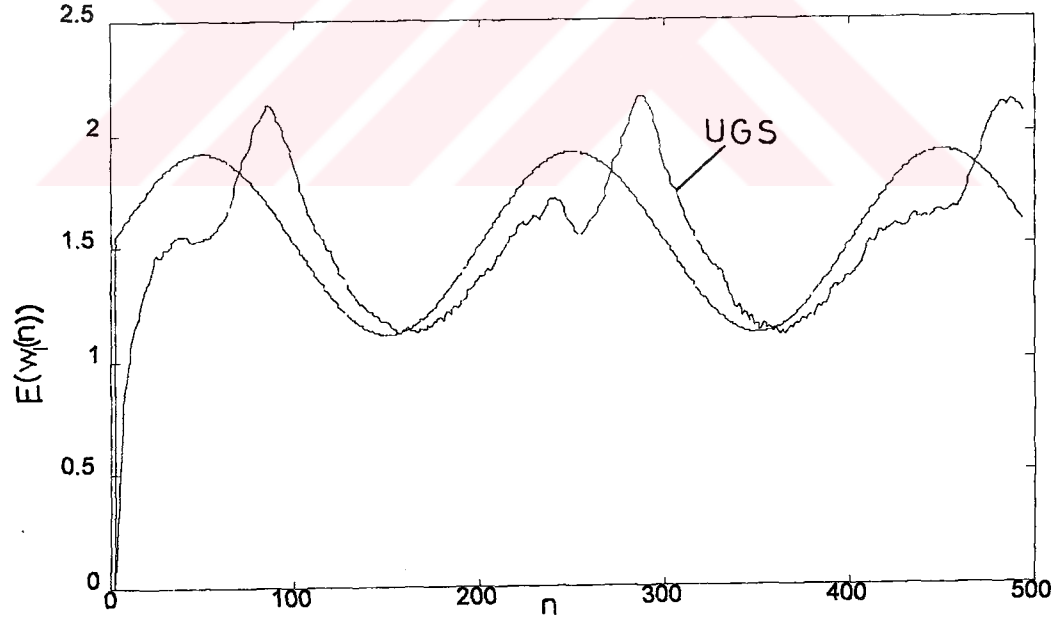
Şekil 4.13

$w_1(n) = 1.5114 + 0.4\ \mathrm{Sin}(2\pi\ 0.005\ n)$, $w_2(n) = -0.95$, $\lambda_R = 0.95$



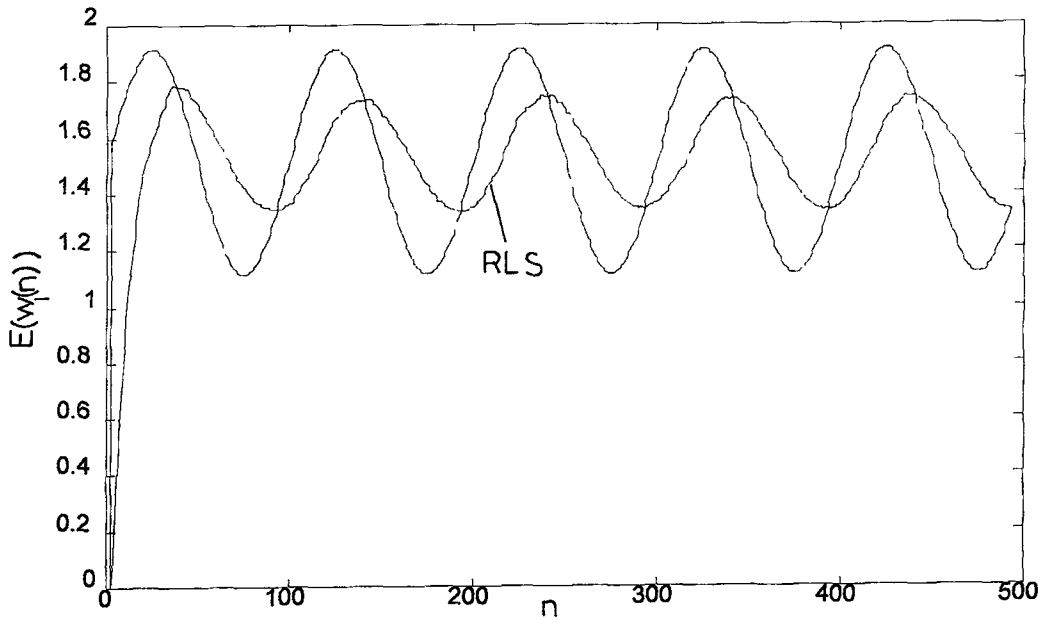
Şekil 4.14

$w_1(n) = 1.5114 + 0.4\ \mathrm{Sin}(2\pi\ 0.005\ n)$, $w_2(n) = -0.95$, $\lambda_G = 0.95$



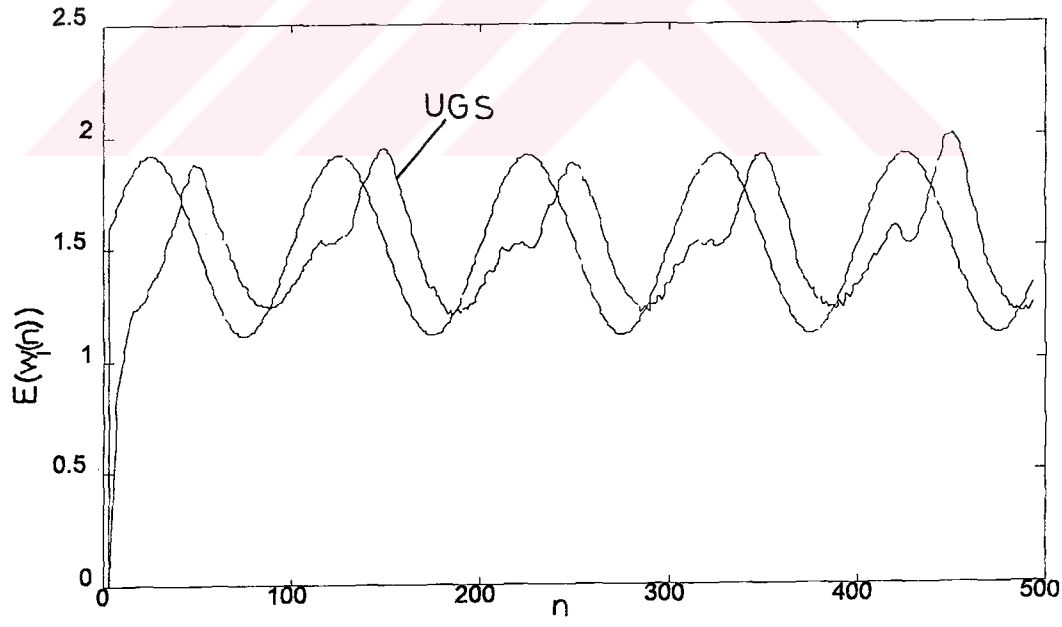
Şekil 4.15

$w_1(n) = 1.5114 + 0.4\ \mathrm{Sin}(2\pi\ 0.005\ n)$, $w_2(n) = -0.95$, $\lambda_G = 0.90$



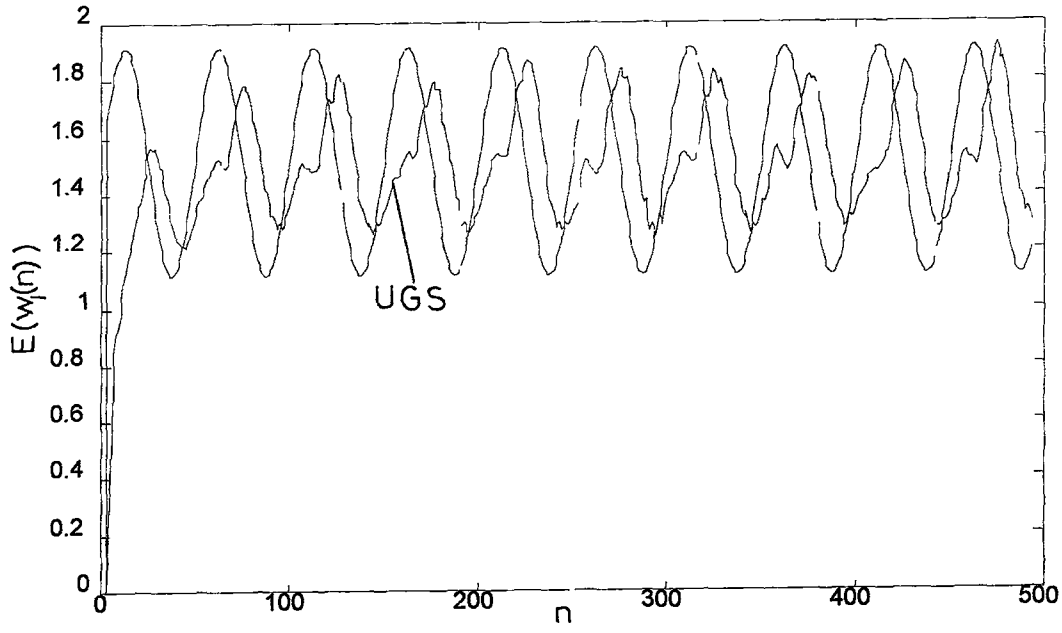
Şekil 4.16

$w_1(n) = 1.5114 + 0.4\ \mathrm{Sin}(2\pi\ 0.01\ n)$, $w_2(n) = -0.95$, $\lambda_R = 0.95$



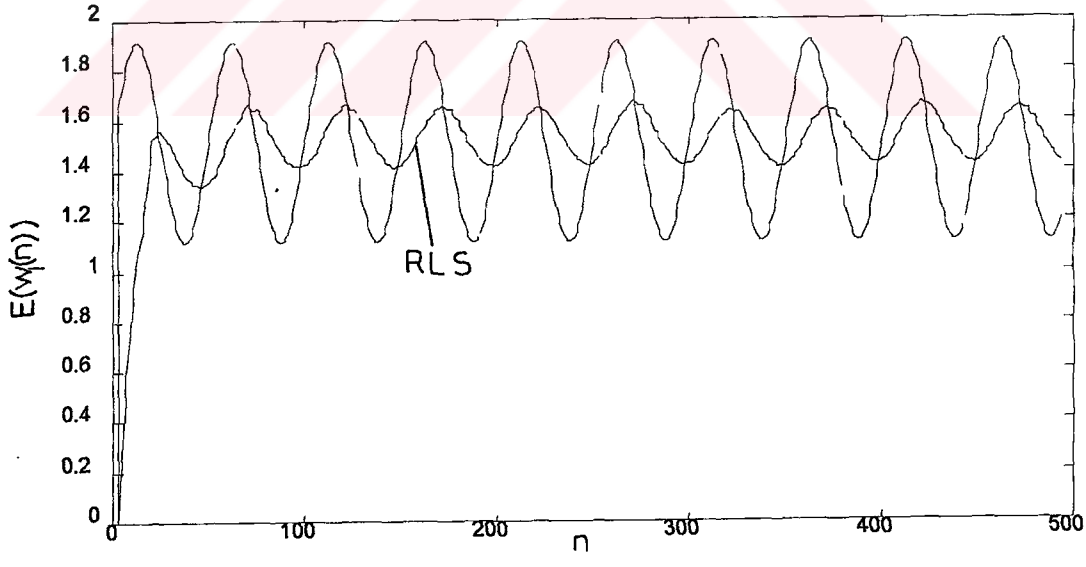
Şekil 4.17

$w_1(n) = 1.5114 + 0.4\ \mathrm{Sin}(2\pi\ 0.01\ n)$, $w_2(n) = -0.95$, $\lambda_G = 0.90$



Şekil.4.18

$w_1(n) = 1.5114 + 0.4\ \mathrm{Sin}(2\pi\ 0.02\ n)$, $w_2(n) = -0.95$, $\lambda_G = 0.85$



Şekil 4.19

$w_1(n) = 1.5114 + 0.4\ \mathrm{Sin}(2\pi\ 0.02\ n)$, $w_2(n) = -0.95$, $\lambda_R = 0.95$

Şekil.(4.13) - şekil.(4.19) da farklı frekanslarda ve farklı ağırlaştırma katsayıları kullanılarak RLS ve Uyarlamalı Gauss-Seidel algoritmaları için izleme performansları incelenmiştir. Beklenen bir sonuç olarak , Uyarlamalı Gauss-Seidel algoritması için oransal özdeğer yaygınlığının büyük olduğu durumlarda daha kötü bir izleme ile karşılaşılmıştır. Bu kötüleşme , ağırlaştırma katsayısına uygun değerler verilerek kısmi olarak giderilmiştir. Frekans arttıkça RLS algoritmasının izleme performansında da bozulma olmaktadır. Ancak RLS algoritması için izleme performansını arttırmak amacıyala ağırlaştırma katsayısı üzerinde önemli değişiklikler yapılamamaktadır.

Sonuç olarak , oransal özdeğer yaygınlığının çok büyük olmadığı durağan olamayan ortamlarda Uyarlamalı Gauss-Seidel ve RLS algoritmaları için aynı performansı gösterdikleri söylenebilir. Ancak AR modeldeki katsayıların değişim hızı arttıkça RLS algoritmasının izleme performansında kötüleşme olmaktadır. Uyarlamalı Gauss-Seidel algoritmasında da görülen bu kötüleşme ağırlaştırma katsayısına uygun değerler verilerek giderilebilmektedir. RLS algoritmasında ağırlaştırma faktörünün küçültülmesi kararsızlık problemine sebep olamaktadır.

4.5 İşlem (madpr) Sayısı Karşılaştırması

LMS algoritması için 2.3.1 bölümünde süzgeç uzunluğu M olmak üzere madpr = 2M olduğu görülmüştür. LMS de madpr süzgeç uzunluğuyla doğrusal olarak artmaktadır. RLS algoritmasında ise madpr süzgeç uzunluğuyla karesel olarak artmaktadır [1] . Özilişki matrisinin simetrik yapısından yararlanarak karesel artan *madpr* sayısı hızlı ardışıl algoritmalarda (Fast RLS) süzgeç uzunluğuyla doğrusal artan hale getirilebilmektedir [23],[24],[25].

Uyarlamalı Gauss - Seidel algoritmasında madpr sayısı bulunmak istenirse (4.10) , (4.11) ve (4.12) eşitliklerinden yararlanılabilir. Algoritma için matris formundaki (4.9) eşitliği de geçerlidir. Ancak bu eşitlik kullanılırsa her adımda altüçgen ve diagonal özilişki matrislerinden oluşan (R_L (n) + R_D (n)) matrisinin tersinin hesaplanması gerekecektir. Bunun dışındaki başka matris ve

vektör çarpma işlemleri de *madpr* yi arttırmaktadır. Programlama açısından basitliği de göz önünde bulundurulursa süzgeç katsayılarının ardışıl hesaplanmasında (4.10) eşitliğinin kullanılması uygun olur. Matris formundaki yapı , (3.6) bölümünde belirtildiği gibi kararlılık incelemesi bakımından bir kolaylık sağlamaktadır.

Süzgeç uzunluğu veya özilişki matrisi boyutu M olan durumda (4.10) eşitliğine göre herhangi bir katsayının hesaplanmasında (M-1) çarpma , toplama ve bir tane bölme işlemi gerekmektedir. Özilişki katsayılarının bilindiği veya hesaplanmış olduğu varsayımıyla her bir iterasyon adımı için bir süzgeç katsayısının hesaplanmasındaki *madpr* sayısı M olur. Süzgeç katsayıları da M tane olduğuna göre bir iterasyonda bütün katsayıların hesaplanması için gerekli madpr sayısı M^2 olmaktadır. Özilişki katsayılarının her adımda yeni durumlarına uyarlanmaları için bir çarpma ve bir toplama işlemi gerekir. Toplam M tane özilişki katsayısı ve M tane çapraz ilişki katsayısı olduğuna göre *madpr* sayısı 2M olmaktadır. Böylece (4.10) eşitliğiyle verilmiş olan Uyarlamalı Gauss - Seidel algoritması için

$$madpr = M^2 + 2M \tag{4.44}$$

olarak bulunur. Uyarlamalı süzgecin değişik uygulama alanlarına göre, eğer istenen işaret d(n) yerine ilgilenilen işaret x(n) kullanılıyorsa çapraz ilişki katsayıları bulunmayacak , fakat özilişki katsayılarının sayısı (M-1) değil M olacaktır. Böyle bir uygulamada Uyarlamalı Gauss - Seidel algoritması için

$$madpr = M^2 + M \tag{4.45}$$

olarak bulunur. RLS algoritmasında *madpr* = ($3M^2$ + 11M + 8) , hızlı algoritmalarda (FRLS) *madpr* = (7m + 16) olmaktadır [1]. Süzgeç uzunluğu M ile madpr nin RLS , FRLS ve Uyarlamalı Gauss - Seidel algoritmaları için değişimi tablo(4.1) de verilmiştir.

Tablo 4.1

M	madpr (***RLS***)	madpr (***FRLS***)	madpr (***UGS***)
1	22	23	2
2	42	30	6
3	68	37	12
4	100	44	20
5	138	51	30
6	182	58	42
7	232	65	56
8	288	72	72
9	350	79	90
10	418	86	110

Görüldüğü gibi Uyarlamalı Gauss - Seidel algoritması madpr karşılaştırmasında RLS e göre her durum , FRLS e göre ise süzgeç uzunluğu M=9 a kadar olan durumlarda üstünlük sağlamaktadır.

4.6 Yakınsama Hızı İncelemesi

Uyarlamalı algoritmalarda , her algoritma adımı için ağırlık katsayıları birer raslantı değişkeni olduğundan yakınsama hızı incelemesinin değişkenlerin ortalamaları üzerinden yapılması uygundur. Her bir ağırlık katsayısının ortalamasının incelenmesi algoritmaların karşılaştırılmasında uygun bir ölçüt değildir. Bunun yerine hata vektörü normunun ortalamasının incelenmesi yerinde olur [26].

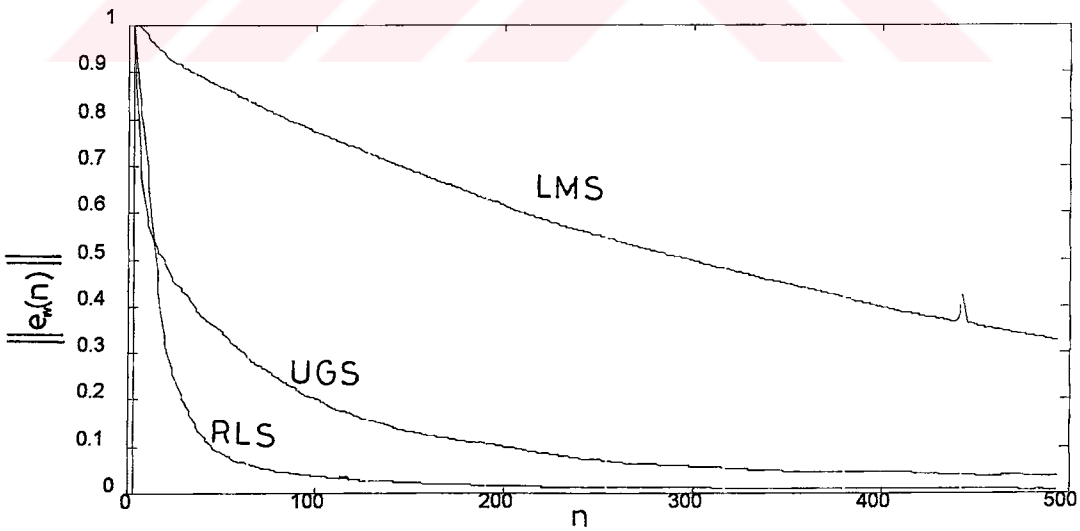
$$\| e_w(n) \| = \| w(n) - w_0 \| \qquad (4.46)$$

eşitliğiyle elde edilen hata vektörünün normu da bir raslantı değişkenidir. Bu raslantı değişkeni optimum katsayı vektörüyle normalize edilir ve ortalaması alınırsa

$$k(n) = \frac{E \| w(n) - w_0 \|}{\| w_0 \|} \qquad (4.47)$$

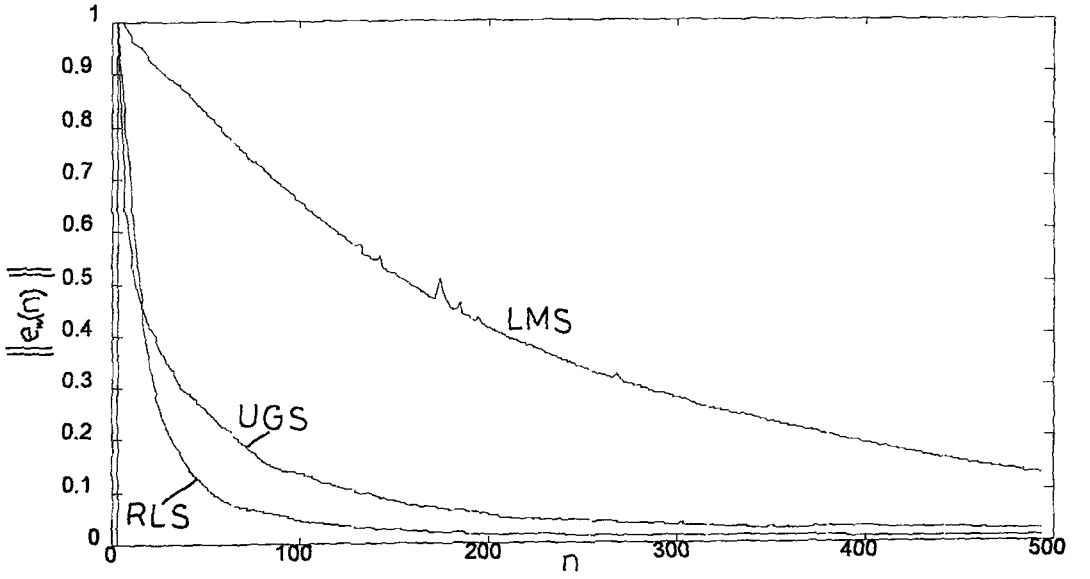
eşitliğiyle tanımlanan ve uyarlamalı algoritmaların yakınsama hızlarının karşılaştırılmasına olanak veren bir ölçüt elde edilir.

Farklı oransal özdeğer yaygınlığına sahip özilişki matrisleri için (4.47) eşitliğiyle tanımlanmış olan ölçüt kullanılarak LMS , RLS ve Uyarlamalı Gauss - Seidel algoritmaları yakınsama hızları karşılaştırmaları şekil (4.20) - (4.23) de verilmiştir.



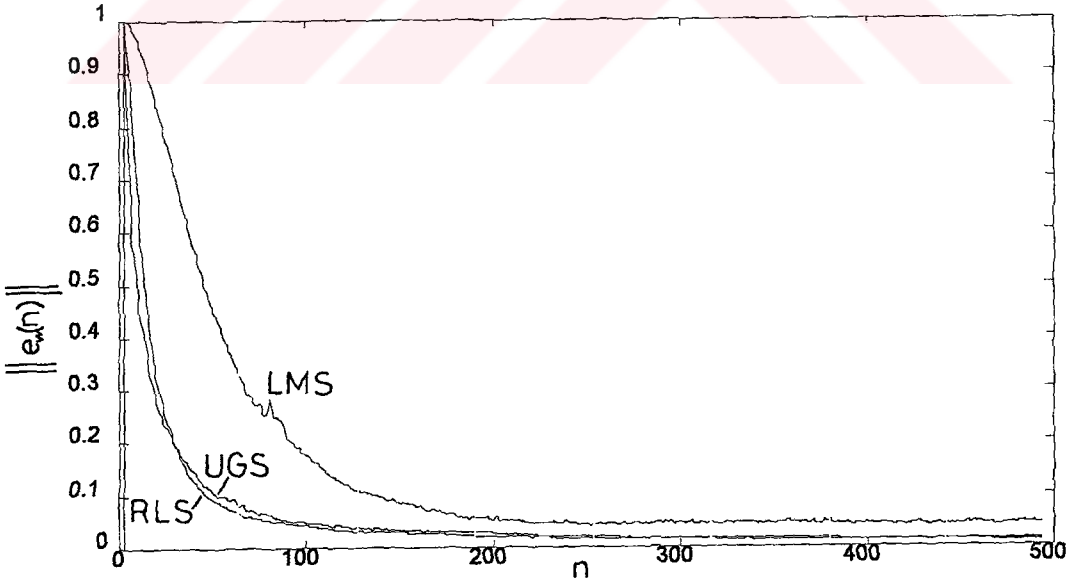
Şekil.4.20

$w_1 = 1.9114$, $w_2 = -0.95$, $X(R) = 100$, $\mu_{LMS} = 0.05$



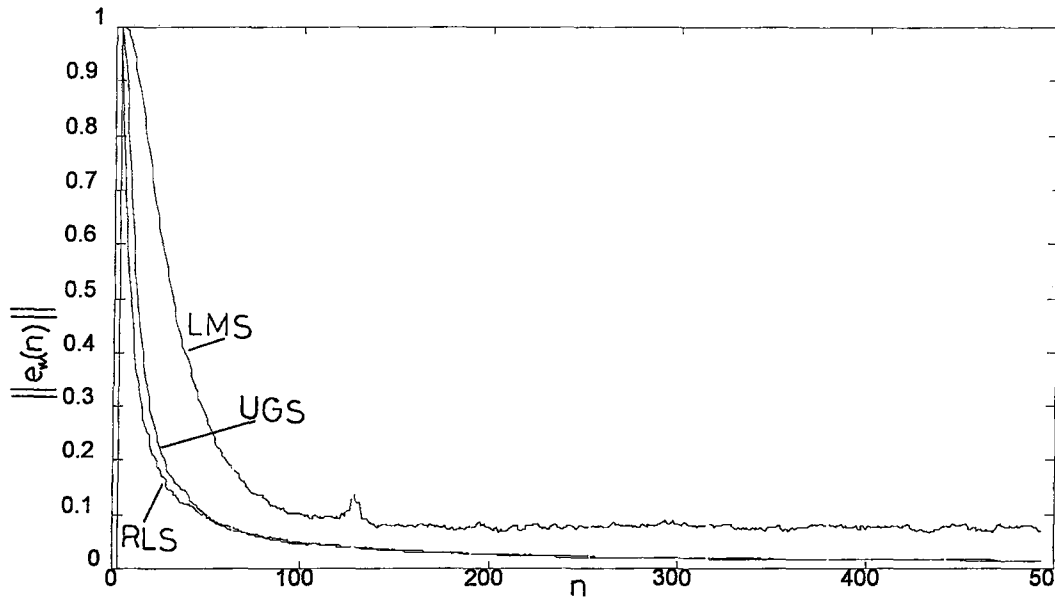
Şekil.4.21

$w_1 = 1.8735$, $w_2 = -0.95$, $X(R) = 50$, $\mu_{LMS} = 0.1$



Şekil.4.22

$w_1 = 1.5955$, $w_2 = -0.95$, $X(R) = 10$, $\mu_{LMS} = 0.5$



Şekil.4.23

$w_1 = 0.9750$, $w_2 = -0.95$, $X(R) = 3$, $\mu_{LMS} = 0.8$

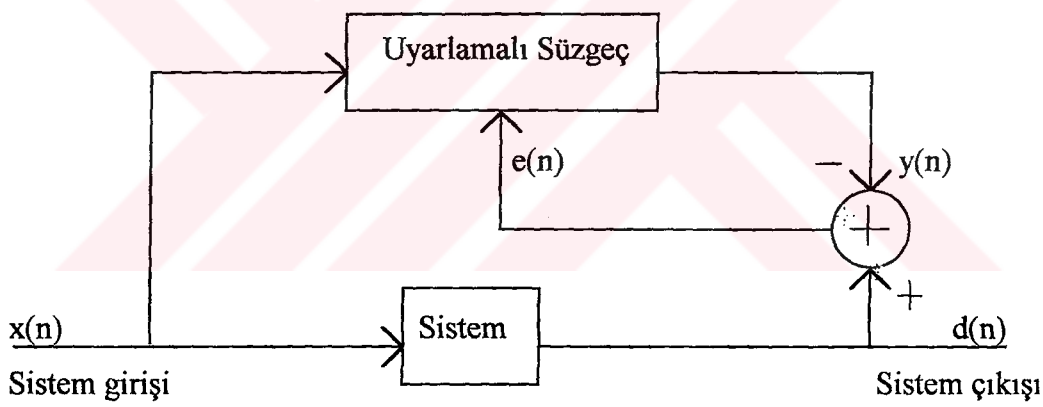
Yakınsama hızı , Uyarlamalı Gauss - Seidel algoritmasında da LMS de olduğu gibi özilişki matrisinin yapısına bağımlıdır. LMS ile yakınsama hızı karşılaştırılması yapıldığında şekil (4.20) - (4.23) dan anlaşılacağı gibi Uyarlamalı Gauss - Seidel algoritmasının daha hızlı olduğu görülür. RLS ile yapılan karşılaştırmada ise oransal özdeğer yaygınlığının büyük olduğu durumda , şekil (4.20) , RLS in daha hızlı olduğu anlaşılmaktadır. Şekil (4.21) - (4.23) incelenirse oransal özdeğer yaygınlığı küçüldükçe RLS ile Uyarlamalı Gauss - Seidel algoritması yakınsama hızlarının yaklaşık aynı kaldığı söylenebilir. Oransal özdeğer yaygınlığının X(R)=10 ve X(R)=3 olduğu , şekil(4.22) ve şekil(4.23) de Uyarlamalı Gauss - Seidel algoritmasının yakınsama hızının RLS algoritmasının yakınsama hızı ile aynı kaldığı görülebilir. Oransal özdeğer yaygınlığının X(R)=50 olduğu şekil(4.21) de , LMS ile yapılan göreceli karşılaştırmada , RLS ve Uyarlamalı Gauss-Seidel algoritmalarının hızlarının yaklaşık olarak eşit olduğu söylenebilir.

BÖLÜM 5

SİSTEM TANILAMA UYGULUMASI

5.1 Sistem Tanılama Uygulaması

Uyarlamalı süzgeçlerin önemli bir uygulama alanı sistem tanılamadır [27],[28] [29],[30]. Sistem tanılama işlemi , sistem modelinin seçilmesi ve bu modeldeki bilinmeyen parametrelerin öngörülmesi şeklinde iki kısımdan oluşur. Uyarlamalı süzgeç kullanılarak sistem tanılama işlemine ait bir blok diagramı şekil (5.1) de verilmiştir.



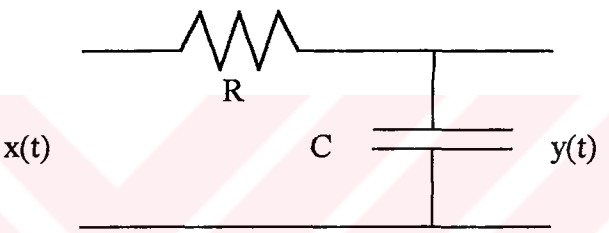
Şekil 5.1

Sistem girişi x(n) , süzgeç katsayıları w_k (n) ve süzgeç uzunluğu M olmak üzere y(n) işareti

$$y(n) = \sum_{k=0}^{M-1} w_k (n)\, x(n-k) \qquad (5.1)$$

eşitliğiyle hesaplanır. Süzgeç çıkışı y(n) ile sistem çıkışı d(n) arasındaki fark e(n) , uyarlamalı algoritmayı kontrol ederek y(n) nin d(n) e yaklaşmasını sağlar. Hatanın karesel ortalamasını minimum yapacak şekilde çalışan algoritma ile sistemin değeri bilinmeyen parametreleri öngörülür.

Bu çalışmada sistem modeli belli olan birinci ve ikinci dereceden sonlu dürtü cevaplı sistemlerin parametreleri RLS ve Uyarlamalı Gauss - Seidel algoritmaları kullanılarak öngörülmüştür. Farklı algoritmalar kullanılarak elde edilen öngörü değerleriyle algoritmalar arasında yakınsama hızı karşılaştırması yapılmıştır.



Şekil 5.2

Şekil (5.2) deki birinci dereceden sistemin girişi x(t) , çıkışı y(t) arasındaki bağıntı

$$x(t) = RC\frac{d\,y(t)}{dt} + y\,t) \qquad (5.2)$$

şeklindedir. Sürekli zamandan ayrık zamana geçişte T örnekleme peryodu olmak üzere türev operatörü yerine geriye doğru fark (backword dif.) [31],[32] yöntemini kullanarak

$$\frac{d\,y(t)}{dt} \cong \frac{y(nT) - y((n-1)T)}{T} \qquad (5.3)$$

yaklaşık eşitliği elde edilir. (5.3) , (5.2) de yerine yazılarak (5.2) yeniden düzenlenirse sistem girişi ile çıkışı arasındaki ayrık zaman fark denklemi bulunur.

$$y(nT) = \frac{RC}{RC+T}\, y((n-1)T) + \frac{T}{RC+T}\, x(nT) \qquad (5.4)$$

(5.4) eşitliğindeki fark denkleminde

$$w = \frac{RC}{RC+T} \qquad a = \frac{T}{RC+T}$$

yazılarak

$$y(n) = w\, y(n-1) + a\, x(n) \qquad (5.5)$$

sonucuna varılabilir. (5.5) eşitliğindeki fark denkleminde x(n) giriş işareti yerine beyaz gürültü uygulanarak LMS , RLS veya Uyarlamalı Gauss - Seidel algoritması işletilmesi sonucu belli bir örnek kümesi için w katsayısı öngörülebilir.

İkinci dereceden bir sistem elde etmek amacıyla şekil (5.2) deki devreden yararlanarak şekil (5.3) deki yapı elde edilebilir. Örnekleme peryodu T olmak üzere , (5.3) eşitliğindeki geriye doğru fark operatörü kullanılarak şekil (5.3) deki sisteme ait fark denklemi

$$w_1 = \frac{2R_1 R_2 C_1 C_2 + T(R_1 C_1 + R_2 C_2)}{R_1 R_2 C_1 C_2 + T(R_1 C_1 + R_2 C_2) + T^2}$$

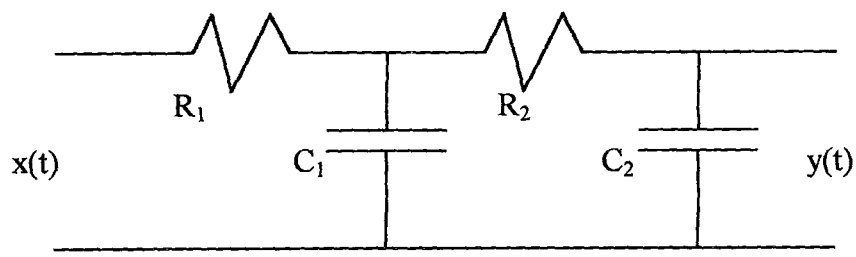
$$w_2 = \frac{- R_1 R_2 C_1 C_2}{R_1 R_2 C_1 C_2 + T(R_1 C_1 + R_2 C_2) + T^2}$$

$$v(n) = \frac{T^2}{R_1 R_2 C_1 C_2 + T(R_1 C_1 + R_2 C_2) + T^2} x(n)$$

olmak üzere

$$y(n) = w_1 y(n-1) + w_2 y(n-2) + v(n) \qquad (5.6)$$

eşitliğiyle elde edilir. (5.6) eşitliğinde v(n) yerine beyaz gürültü uygulanarak herhangi bir uyarlamalı algoritma yardımıyla belli bir örnek kümesi üzerinden w_1 , w_2 katsayıları için öngörü yapılabilir.



Şekil 5.3 İkinci dereceden sistem

5.2 Simulasyon Sonuçları

Şekil (5.2) deki sistemin girişine beyaz gürültü uygulanarak (5.5) eşitliğindeki

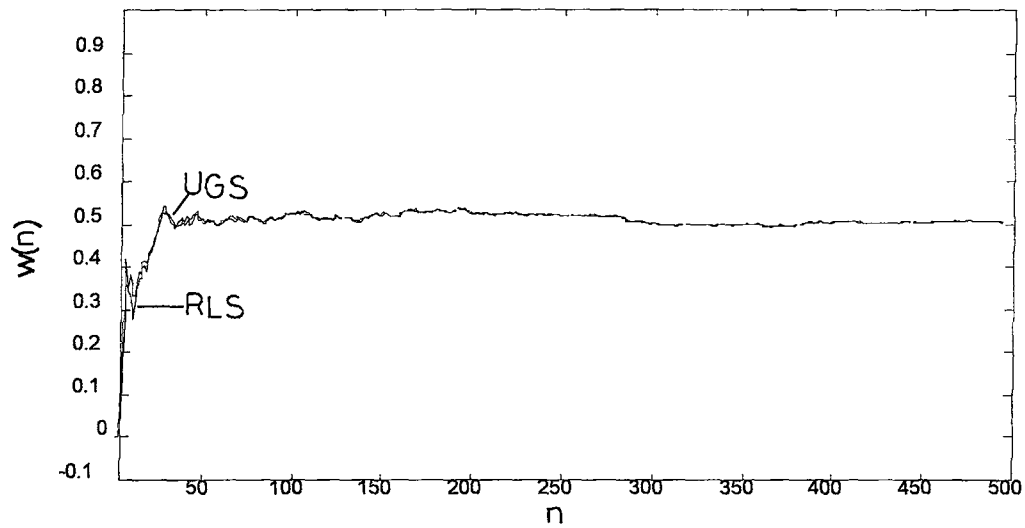
$$w = \frac{RC}{RC + T}$$

katsayısı 500 örneklik kümeler kullanılarak RLS ve Uyarlamalı Gauss - Seidel algoritmalarıyla öngörülmüştür. Birbirinden bağımsız deneme sayısı 100 deneme üzerinden hesaplanan öngörü ortalamasının değişimi şekil (5.4) de verilmiştir. Şekil (5.3) deki devrenin girişine beyaz gürültü uygulanarak (5.6) eşitliğindeki

$$w_1 = \frac{2R_1R_2C_1C_2 + T(R_1C_1 + R_2C_2)}{R_1R_2C_1C_2 + T(R_1C_1 + R_2C_2) + T^2}$$

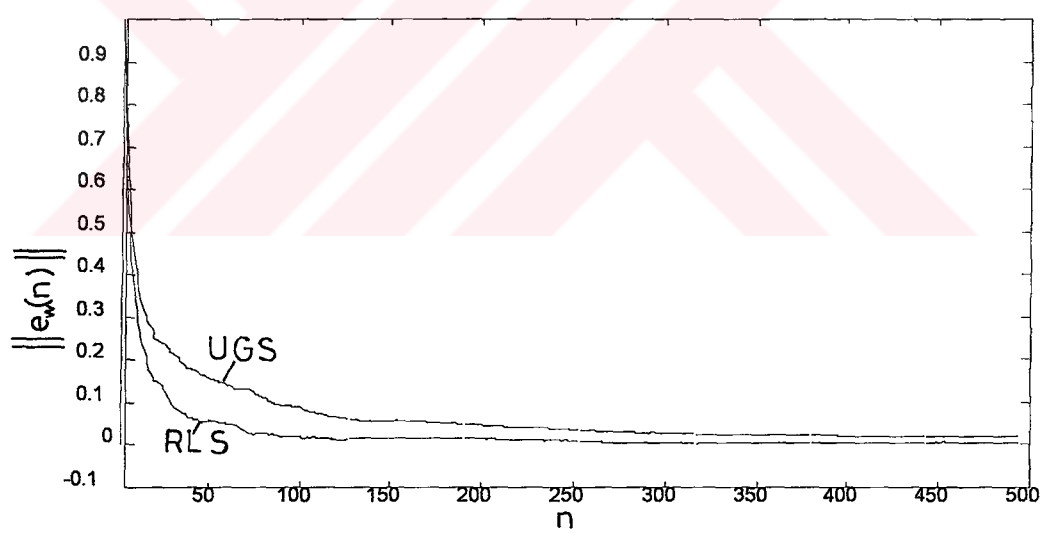
$$w_2 = \frac{-R_1R_2C_1C_2}{R_1R_2C_1C_2 + T(R_1C_1 + R_2C_2) + T^2}$$

katsayıları R_1 , R_2 , C_1 ve C_2 nin değişik değerleri için 500 örneklik kümeler kullanılarak RLS ve Uyarlamalı Gauss - Seidel algoritmalarıyla öngörülmüştür. Birbirinden bağımsız deneme sayısı 100 için (4.46) eşitliğiyle hesaplanan katsayı vektörü normu ortalamaların değişimleri şekil (5.5) - şekil(5.6) da verilmiştir.



Şekil 5.4

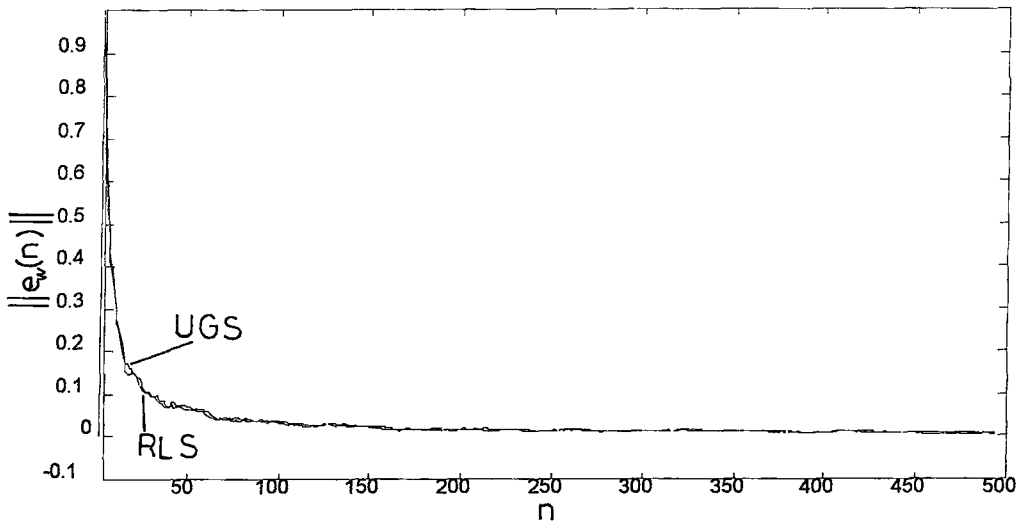
$R = 1k\Omega$, $C = 0.1\mu F$, $T = 0.1ms$, $w = 0.5$



Şekil 5.5

$R_1 = 1k\Omega$, $R_2 = 10k\Omega$, $C_1 = 0.1\mu F$, $C_1 = 0.1\mu F$, $T = 0.1ms$

$w_1 = 1.4091$, $w_2 = -0.4545$



Şekil 5.6

$R_1 = 1k\Omega$, $R_2 = 1k\Omega$, $C_1 = 0.1\mu F$, $C_1 = 0.1\mu F$, $T = 0.1ms$

$w_1 = 1$, $w_2 = -0.25$

Birinci dereceden sistemler için katsayı vektörü yerine bir skaler , w katsayısı , özilişki matrisi yerine de yine bir skaler olan r_0 (n) , işaretin varyansı , geldiği için RLS ve Uyarlamalı Gauss - Seidel algoritmalarının sonuçları şekil (5.4) den görüldüğü gibi aynı olmaktadır. İkinci dereceden sistemlerde ise oransal özdeğer yaygınlığının büyük olduğu şekil(5.5) de RLS in yakınsama hızı Uyarlamalı Gauss - Seidel algoritmasınınkine göre daha büyüktür. Oransal özdeğer yaygınlığı küçüldükçe şekil(5.6) dan görüleceği gibi iki algoritmanın yakınsama hızlarının yaklaşık olarak aynı kaldığı söylenebilir. Böyle durumlarda

SONUÇLAR

Uyarlamalı kontrol algoritmalarının birbirleriyle karşılaştırılmasında kullanılan ölçütlerden önemli iki tanesi algoritmanın optimum çözüme ulaşması için gerekli olan iterasyon sayısıyla ilişkili olan *yakınsama hızı* ve bir iterasyon adımındaki çarpma ve bölme işlemi sayısı (*madpr*) dır. Doğal olarak algoritmalardan istenen *yakınsama hızının* büyük , *madpr* sayısının küçük olmasıdır. Gradient tabanlı algoritmalar da *yakınsama hızı* ardışıl algoritmalarınkine göre daha küçüktür. Yakınsama hızın büyük olması işlem sayısında da artışa sebep olmaktadır. Süzgeç uzunluğu M olmak üzere , *madpr* sayısı M ile doğrusal artan algoritmalar $O(M)$, karesel artan algoritmalar $O(M^2)$ sınıfından algoritmalar olarak adlandırılmaktadır. Gradient tabanlı bir algoritma olan LMS $O(M)$, ardışıl bir algoritma olan RLS ise $O(M^2)$ sınıfından algoritmalardır. Hızlı ardışıl algoritmalar (FRLS) , $O(M)$ sınıfından algoritmalar olmalarına rağmen yakınsama hızı bakımından RLS ile aynı istatistiksel özellikleri göstermektedirler.

Bu çalışmada önerilen Uyarlamalı Gauss-Seidel algoritması da $O(M^2)$ sınıfından bir algoritmadır. Bu bakımdan *madpr* sayısı karşılaştırılmasında ilk bakışta yalnız RLS algoritmasına göre bir üstünlük sağlayabileceği söylenebilir. Gerçekten de Uyarlamalı Gauss-Seidel ve RLS algoritmalarının *madpr* sayısı karşılaştırmaları yapıldığında süzgeç uzunluğunun her değeri için Uyarlamalı Gauss-Seidel algoritmasında *madpr* sayısının daha küçük olduğu görülür. Aynı karşılaştırma Uyarlamalı Gauss-Seidel ve FRLS algoritmaları arasında yapıldığında , FRLS algoritması $O(M)$ sınıfından bir algoritma olmasına rağmen , süzgeç uzunluğu M=7 durumuna kadar Uyarlamalı Gauss-Seidel algoritmasında *madpr* sayısı daha küçük olmaktadır. Uyarlamalı Gauss-Seidel algoritmasının , FRLS algoritmasına göre *madpr* sayısı karşılaştırmasında sağladığı bu üstünlük özel bir sistem tanılama uygulamasında veya genel olarak çapraz ilişki vektörünün bulunmadığı yalnız özilişki katsayılarının bulunduğu uyarlamalı işaret işleme uygulamalarında süzgeç uzunluğu M=8 durumu için de geçerli olmaktadır.

Değişik oransal özdeğer yaygınlığına sahip özilişki matrisleri üzerinde yapılan çalışmalarda Uyarlamalı Gauss-Seidel algoritmasının yakınsama hızının LMS algoritmasının yakınsama hızına göre daha büyük olduğu görülmüştür. Gradient tabanlı algoritmalarda olduğu gibi Uyarlamalı Gauss-Seidel algoritmasında da yakınsama hızı özilişki matrisinin oransal özdeğer yaygınlığına bağımlıdır. LMS algoritmasında da karşılaşılan , oransal özdeğer yaygınlığının yakınsama hızına olan bu olumsuz etkisi Uyarlamalı Gauss-Seidel algoritmasında daha az olmaktadır. Yakınsama hızı karşılaştırmasında Uyarlamalı Gauss-Seidel algoritmasının LMS algoritmasına göre başka bir üstünlüğü de adım parametresiyle ilgilidir. Bilindiği gibi gradient tabanlı algoritmalarda adım parametresi algoritmanın kararlılığına ve yakınsama hızına etkimektedir. Algoritmanın kararlı olması için adım parametresinin içinde bulunması gereken aralığın belirlenmesi LMS algoritmasının bir sakıncasıdır. Yakınsama hızı bakımından adım parametresinin bu aralık içindeki uygun değerinin seçilmesi de başka bir problemdir. Uyarlamalı Gauss-Seidel algoritmasında kararlılık ve yakınsama hızı bakımından , LMS algoritmasında görülen bu sakıncalar bulunmamaktadır.

Uyarlamlı Gauss-Seidel algoritmasının RLS ve FRLS algoritmalarına göre diğer bir üstünlüğü de programlamaya uygun basit yapısıdır. Uyarlamalı Gauss-Seidel algoritmasında herhangi bir süzgeç katsayısının bir sonraki algoritma adımındaki değeri hesaplanırken ilgilenilen katsayının bulunulan algoritma adımındaki değeri kullanılmamaktadır. Bu özellikten de yararlanarak eğer katsayı hesaplamaları birbirini izleyen program satırlarında yapılırsa algoritma adımını veya ayrık zaman değişkenini gösteren indisi kullanma gereği yoktur. Bunun doğal sonucu olarak bellek kullanımı ve erişimi daha az olan basit bir programlama tekniği elde edilir. Programlama tekniğindeki bu farklılık yüksek seviyeli programlama dillerinin kullanıldığı uygulamalarda önemli bir kolaylık getirmese de sayısal işaret işlemcilerin kullanıldığı gerçek zaman uygulamalarında üstünlük sağlamaktadır.

KAYNAKLAR

[1] HAYKIN, S. , Adaptive Filter Theory, Prentice Hall, 1991.

[2] LUENBERGER, D.G. , Linear And Nonlinear Programming, Prentice Hall, 1972.

[3] WIDROW, B. , HOFF, M.E. , Adaptive Switching Circuits,IRE WESCON Conv. Rec. pt. 4, pp.96-104, 1960.

[4] WIDROW, B. , Adaptive Filters, in Aspects of Network and System Theory, ed. R.E.Kalman and N.De Claris, NewYork, 1970.

[5] WIDROW, B. , STEARNS, S.D. , Adaptive Signal Processing, Prentice Hall, 1985.

[6] CARAYANNIS,G. ,MANOLAKIS,D.G. ,KALOUPTSIDIS,N. ,A Fast Sequential Algorithm for Least-Squares Filtering and Prediction, IEEE Transactions on ASSP. vol.31,pp.1394-1402 , 1983.

[7] FISHER,B. ,BERSHAD,N.J. , The Complex LMS Adaptive Algorithm Transient Weight Mean and Covariance with Applications to the ALE, IEEE Transactions on ASSP. vol.31,pp.34-44 , 1983.

[8] BITMEAD,R., ANDERSON,B.D. , Performance of Adaptive Estimation Algorithms in Dependent Random Environments, IEEE Transactions on AC.vol.25,pp.788-794 , 1980.

[9] KENNETH,R.P. ,A Distributed-μ Implementation of the LMS Algorithm, IEEE Transactions on ASSP. vol.29,pp.753-762 , 1981.

[10] BERSHAD,N.J. ,FEINTUCH,P.L. ,A Normalized Frequency Domain LMS Adaptive Algorithm, IEEE Transactions on ASSP. vol.34,pp.452-461 , 1986.

[11] REN,Z. ,SCHÜTZE,H. ,HARTMANN,I. ,Modified Fast Exact NLMS Adaptive Algorithm, Electronics Letters ,vol.30,pp.1029-1031,1994.

[12] DIRK,T.M. , On the Converge Behaivor of the LMS and the Normalized LMS Algorithms , IEEE Transactions on Signal Processing , vol.41 pp.2811-2825 , 1993.

[13] KAILATH,T. ,An Innovations Approach to Least-Squares Estimation:Part I. Linear Filtering in Additive White Noise, IEEE Transactions on AC,vol.13,pp.646-655 , 1968.

[14] HO,Y.C. , On the Stochastic approximation Method and Optimal Filter Theory, J. Math. Anal. Appl.,vol.6,pp.152-154 , 1963.

[15] BROOKS,L.W. ,REED,I.S. , Equivalence of the Likelihood Ratio Processor the Maximum Signal-to-Noise Ratio Filter and the Wiener Filter, IEEE Trans. Aerospace and Electron. Syst., vol.8,pp.690-692, 1972.

[16] JOHNSON,L.W. ,RIESS,R.D. , Numerical Analysis, Addison-Wesley, pp.68-72 , 1982.

[17] BURDEN,R.L. ,FAIRES,J.D. , Numerical Analysis, PWS Publishers, pp.424-430 , 1985.

[18] DAHLQUIST,G. ,BJÖRCK,A. , Numerical Methods, McGraw Hill , pp.188-193 , 1974.

[19] VARGA,R.S. , Matrix Iterative Analysis, Prentice-Hall, pp.322-327, 1962.

[20] GOLUB,G.H. , VANLOAN,C.F. , Matrix Computations, North Oxford Academic Press. , pp.353-358 , 1988

[21] FISZ,M. , Probability Theory and Mathematical Statistics, John Wiley and Sons, Inc. , pp.358-363 , 1963.

[22] PETERS,S.D. , ANTONIOU,A. , A Parallel Adaptation Algorithm for Recursive-Least-Squares Adaptive Filters in Nonstationary Environments IEEE trans. Signal Processing, vol.43, pp.2484-2494, 1995.

[23] MANOLAKIS,D.G. ,KALOUPTSIDIS,N. ,CARAYANNIS,G. , Fast Algorithms for Discrete Time Wiener Filters with Optimum Lag, IEEE transactions on ASSP, vol.31, pp.168-179 , 1983.

[24] AKAIKE,H. , Block Toeplitz Matrix Inversion, SIAM J. Appl. Math. vol.24, pp.234-241, 1973

[25] CARAYANNIS,G. , KALOUPTSIDIS,N. , MANOLAKIS,D.G. , Fast Recusive Algorithms for a Class of Linear Equations, IEEE transactions on ASSP, vol.30, pp.227-239 , 1982.

[26] CHAMBERS,J.A. ,TANRIKULU,O. ,CONSTANTİNİDİS,A.G. , Least Mean Mixed-Norm Adaptive Filtering, Electronics Letters, vol.30, pp.1574-1575 , 1994.

[27] ASTRÖM,K.J. ,WITTENMARK,B. , Computer Controlled Systems, Prentice Hall, 1990.

[28] SÖDERSTROM,T. ,STOICA,P. , System Identification, Prentice Hall 1989.

[29] LJUNG,L. , System Identification: Theory for the User, Prentice Hall 1987.

[30] GOODWIN,G.C. ,PAYNE,R.L. , Dynamic System Identification: Experiment Design and Data Analysis, Academic Press , 1977.

[31] PHILLIPS,C.L. , NAGLE,H.T. , Digital Control System Analysis and Design, Prentice Hall , 1986.

ÖZGEÇMİŞ

Osman Hilmi Koçal 1967 yılında İstanbul'da doğdu. İlk ve orta öğrenimini Yalova'da tamamladı. İ.T.Ü Elektrik-Elektronik Fakültesinden 1988 de mühendis , 1991'de yük. müh ünvanını aldı. 1987 - 1991 yılları arasında özel bir mühendislik bürosunda çalıştı. 1991 yılında İ.T.Ü Elektrik-Elektronik Fakültesi Kontrol-Kumanda Sistemleri Ana Bilim Dalında araştırma görevlisi olarak çalışmaya başladı. Osman Koçal evli ve iki çocuk babasıdır.