YILDIZ TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

Lpc Kullanan Bir Ses Tanıma Sisteteminin Bilgisayar Simülasyonu

Mustafa Turfan

Yüksek Lisans Tezi

YILDIZ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

LPC KULLANAN BİR SES TANIMA SİSTEMİNİN BİLGİSAYAR SİMÜLASYONU

YÜKSEK LİSANS TEZİ

MÜH. MUSTAFA TURFAN

İSTANBUL 1989

YILDIZ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

LPC KULLANAN BİR SES TANIMA SİSTEMİNİN BİLGİSAYAR SİMÜLASYONU

YÜKSEK LİSANS TEZİ

MÜH. MUSTAFA TURFAN

İSTANBUL 1989

İÇİNDEKİLER SAYFA NO

ÖZET

Bu tezde esasen işaret işleme teorisinin inceleme dallarından biri olan ses işaretlerinin işlenmesi konusuna sadece bir giriş yaptım. Çok değişik boyutları olan meseleyi ana hatlarıyla incelediğimiz için benzer çalışma yapmak isteyenlere bir başlangıç kaynağı olur ümidindeyim.

Birinci bölümden üçüncü bölümün sonuna kadar olan kısımda temel filtre tanımlarıyla birlikte LPC filtrelerinin detaylı bir incelenmesi verildi. Dördüncü bölümde insan konuşmasının elektriksel açıdan mekanizması incelendi. Beşinci bölümde logaritmik ses tanıma dediğimiz ve bu tezde de kullandığımız ses tanıma yöntemi anlatıldı. Altıncı bölümde ise bu ses tanıma metodunu kullanarak hazırlanan programın yapısı ile alınan sonuçlar detaylarıyla verildi. Programın kendisi ise kitabın sonuna EK-I olarak ilave edildi.

Gerek tez çalışmaları sırasında gerekse kitabın hazırlanması esnasında başta Doç.Dr. Metin YÜCEL olmak üzere okulda hocalarımın, işyerinde de mesai arkadaşlarımın gerçekten samimi destek ve anlayışlarını gördüm. Bu vesile ile hepsine ayrı ayrı teşekkür ederim.

SUMMARY

In this these, an introduction has been made to speech processing. It is expected that this overview becomes a starting point for the people who will make a similar work in such a field.

From the beginning of Chapter-I to the end of Chapter-III basic filter definitions and a detailed description of LPC filters are given. In Chapter-IV, mechanism of human speech from an electrical point of view is given. In Chapter V, method of logarithmic speech recognation which has been used in this essay is described. In Chapter VI, structure of the Computer program based on this algorithm and the results are given in detail. The program itself is given in Appendix-I.

The auther wishes to thank to Dr. Metin Yücel for his guidence during the preparation of this thesis, to the staff of the Department of Electronic Engineering and to his collegues for their support and encouragement.

GİRİŞ

Sayısal bilgisayar teknolojisindeki hızlı yükselmeye paralel olarak ayrık işaret işleme teorisi de uygulamada bir çok alanda kendisine kullanım yeri buldu. Bugün üzerinde hala ilgiyle çalışılan işaret işleme konularından biri de ses işaretlerinin analiz ve sentezidir. Bu konudaki çalışmalar genelde iki gruba ayrılabilir. Bunlar ses sentezi de demek olan yapay ses üretimi ile, ses tanımadır. Biz bu çalışmada ses tanıma olayı ile ilgilendik.

Ses tanıma konusuda kendi içinde birçok dallara ayrılan bir konudur. Yani meselenin boyutları epeyce geniştir. Sadece bir kişinin sesinden yalnızca birkaç kelime yada harfi tanıyan basit sistemlerden başlayıp çok daha karmaşık olan sistemlere giden bir akışı vardır. Bugün kişiden bağımsız olarak çalışan ve kütüphanesi de epeyce geniş olan ses tanıma ve anlama sistemleri uygulamada birçok alanda mevcuttur.

Bu çalışmada biz bu sistemlerin en basiti diyebileceğimiz kısmı üzerinde çalıştık. Yani kişiye bağımlı ve kütüphanesi oldukça fakir olanı... Kütüphanede yalnızca bir'den on'a kadar olan rakamların söylenişleri mevcuttur. Yapılan çalışma ise esasen bir simülasyon olup bu kapasiteye sahip bir kütüphanedeki kelimeleri tanıyan bir programdır. Ses işaretine ait dataların incelenmesinde ise LPC filtre analizi kullanılmıştır. Analiz sonucunda tanıma paternleri dediğimiz ve esasen LPC filtresi çıkışındaki

hataların karasel toplamı olan parametreler elde edilmiştir. Bu parametreler kütüphanedeki referans kelime başına bir tanedir. Bu parametreler yardımıyla ise logaritmik bir uzaklık fonksiyonu tanımlanarak bir tanıma metodu geliştirilmiştir. Önümüzdeki bölümlerde genel filtre teorisinden başlayarak programın teşkiline kadar yapılan çalışma adım adım incelenmiştir.

1-SAYISAL FİLTRELER

1.1. Filtreler

Filtreleme terimi genelde fiziki donanım (hardware) ya da yazılım (software) vasıtasıyla, gürültüye ve harici istenmeyen etkilere maruz kalmış bir işaretin içindeki gerçek bilgiyi çıkararak gürültüden arındırma işlemi için kullanılır. Mesela bir haberleşme sisteminde verici taraftan gönderilen bir haber işareti, iletim ortamında çeşitli gürültü etkenlerine maruz kalıp bozulabilir. Bu gürültülü haber işaretinden,gerçek haber işaretini değilse bile,mümkün olduğu kadar benzerini elde etmek için çeşitli filtreleme işlemlerine ihtiyaç duyulur. Filtreleme işleminin başarısı, filtrelemenin cinsine ve kullanılan yöntemlere bağlı olarak geniş bir aralıkta değişir. Elektrik mühendisliğinde filtreler en genelde analog ve sayısal filtreler olmak üzere ikiye ayrılır. Her iki çeşit filtrenin de kendine has uygulama alanı olmakla beraber son yıllarda sayısal bilgisayar teknolojisindeki hem donanım hem yazılım alanında olan süratli gelişmeler en karmaşık işaret işleme algoritmalarını dahi hızlı, kolay ve daha emniyetli bir tarzda uygulama imkanı verdiğinden bugün artık her türlü filtre tasarımı sayısal yöntemlerle yapılabilmektedir. Çeşitli türde sayısal filtreler elektrik-elektronik teknolojisinin uygulandığı geniş alanın her kademesinde görülebilmektedir.

Filtreleri ayrıca çalışma prensiplerine göre lineer-nonlineer ve adaptif-adaptif olmayan filtreler gibi türlere ayırmak da mümkündür. Bunlardan şüphesiz lineer ve adaptif

olmayan filtreler en basit yapıya sahip olanıdır. Ancak sayısal filtrenin bu basit hali bile pratikte çoğu problemleri gidermeye kafi gelir. Filtrenin kullanım yeri ve ihtiyacına göre türüde yukardaki tanımlara uygun olarak değişir. Mesela robotik sistemler gibi karmaşık sistemlerde genelde en karmaşık filtre yapısı olan nonlineer-adaptif filtreler kullanılır. Bu alt grupların herbiri kendi alanında bir uzmanlık sahasını oluşturup zaman zaman oldukça yüklü matematik gerektirirler. Ancak biz bu çalışmada fazla detaylı bir incelemeye girişmeyeceğiz. Bu bizim konumuzun dışındadır. Ses tanıma işleminde kullandığımız filtre esas olarak bir "Lineer Prediktör Filtresi" olup, adından anlaşılacağı üzere lineer ve adaptif olmayan bir filtre yapısındadır. Dolayısıyla nispeten basit yapıya sahiptir. Bu filtrelerle ilgili temel bilgiler ise 2.ve3.bölümlerde detaylarıyla verilecektir. Okuyucunun bu incelemelerde ayrık matematik ve temel elektronik bilgilere sahip olduğu kabul edilmiştir.

Lineer prediktör filtrelerini incelemeye geçmeden önce ilerdeki konularda kullanacağımız bazı terimleri burada kısaca izah edelim.

1.2. Wiener ve Kalman Filtreleri

Filtrelerin kullanım amacının, gürültülü bilgi işaretlerinin, gürültüden arındırılması olduğunu daha önce söylemiştik. Dolayısıyla bu arındırma işleminde bir optimizasyon söz konusudur. Yani bazı filtre parametreleri öyle seçilmelidir ki, filtre çıkışında elde edilen bilgideki hata payı minimum olsun. Bu optimizasyon işlemi bütün filtrelerde karşımıza çıkar ancak, bunun yolu tek değildir. Bu amaçla literatürde birçok optimizasyon yöntemi bulmak mümkündür. İşte bu yöntemlerden biri de "En Küçük Kareler"

yöntemidir. Optimizasyonu bu yöntemle yapılan lineer filtrelere Wiener Filtresi denir. Klasik Wiener teorisinde optimizasyon yapılmakla beraber filtre parametrelerinin zamanla değişmediği, yani üzerinde işlem yapılan işaretin istatiksel olarak durağan olduğu kabul edilmiştir. Ancak pratikte daha ziyade bunun aksi durumlar söz konusudur. Mesela ses işaretinde böyle bir durum vardır.4.bölümde açıklandığı gibi bir sesli harfe ait konuşma işareti belli pitch peryodları arasında durağan sayılabilmektedir. Ancak aynı harfin başka bir şahıs tarafından konuşulması durumunda ya da aynı şahıs tarafından fakat farklı bir zaman ve ortamda konuşulması halinde bu durağanlık kaybolur. Dolayısıyla bu ses işaretini işleyen filtreye ait katsayılarda, sürekli değişir. Eğer bu değişiklik nazara alınmazsa tasarlanan filtrenin başarısı istenen düzeye ulaşmaz. İşte durağan olmayan işaretlerin işlendiği ve katsayıları zamanla değişen filtrelerin optimizasyon işlemi 1960'lı yıllarda Kalman tarafından incelenmiş olup, bu çeşit filtrelere de Kalman filtreleri denmiştir. Bu çalışmada Kalman Filtreleri incelenmeyecektir.

1.3. (FIR) ve (IIR) Filtreler

Bir sayısal filtrenin giriş ve çıkışı arasındaki ilişkiyi en genelde matematiksel olarak aşağıdaki gibi ifade edebiliriz:

$$y(n) = \sum_{k=o}^{M} a(k).\, u(n-k) + \sum_{k=1}^{N} b(k)\, y(n-k) \qquad (1.1)$$

Burada y(n) çıkış dizisi, u(n) griş dizisi, a(k) ve b(k)'de bunlara ilişkin filtre katsayılarıdır. (1.1)'e "Fark Denklemi"de denir. Bu ifadeye bakıldığında çıkış dizisinin n anındaki değeri, çıkış dizisiyle giriş dizisinin daha önceki anlardaki değerlerinin lineer birleşimlerine eşittir.

Filtrelerin bu genel ifadesinden hareketle iki tür filtreyi tarif etmek mümkündür: FIR ve IIR filtreler.

a) FIR Filtre

b(k) katsayılarının hepsinin sıfır olduğunu kabul edersek y(n) ifadesi sadece giriş dizisine bağımlı kalır:

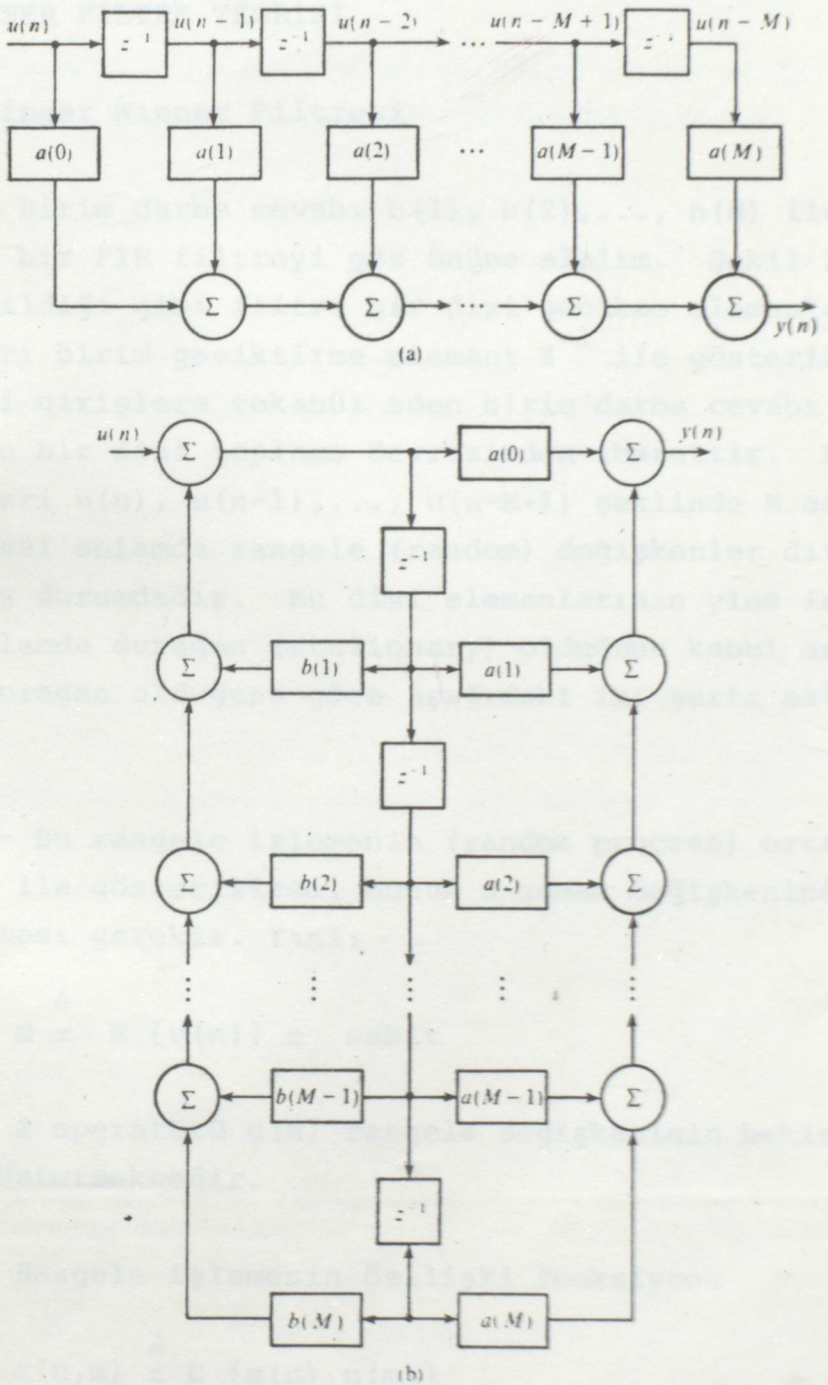
$$y(n) = \sum_{k=0}^{M} a(k)\, u(n-k) \qquad (1.2)$$

Bu ifadeye tekabül eden filtrenin fiziksel yapısı Şekil 1.1.a'da gösterilmiştir. Burada Z^{-1}, birim gecikme elemanını gösterir. Dikkat edilirse böyle bir filtreyi hardware olarak lojik devre elemanlarıyla gerçekleştirmek mümkündür. Z^{-1} geciktirme elemanı, esasen bir ötelemeli yazıcıyla, (shift register) keza toplama ve çarpma blokları da lojik devre tasarımıyla gerçekleştirilebilir. Böyle filtrelere literatürde sonlu darbe cevabı filtresi (Finite Impulse Response) ya da kısaca FIR filtre denir. Bu filtreler yapılarının gereği olarak kararlıdırlar.

b) IIR Filtre

b(k) katsayılarından bir veya bir kaçının sıfırdan farklı olması halinde y(n) çıkış dizisinin belli bir n anındaki değerini, daha evvelki çıkış dizisi değerleri de etkiler. Yani, sistem geri beslemeli hale gelir. Bu yapıya sınırsız darbe cevabı (infinite impulse response) ya da kısaca IIR filtreleri denir. Böyle bir filtre yapısını fiziksel olarak çeşitli yollarla gerçekleştirmek mümkündür. Şekil 1.1 b'de M=N için böyle bir filtrenin gerçeklenmesi gösterilmiştir. IIR filtreler b(k) geribesleme katsayılarına bağlı olarak kararsız da olabilirler. Bu tabiki çoğunlukla istenmeyen bir durumdur. Dolayısıyla gerekli önlemleri

almak gerekir. Bu çalışmada inceleyeceğimiz ve kullanacağımız lineer prediktör filtrelerinin hepsi de FIR filtre yapısındadır.



Şekil 1.1: (a) FIR filtre, (b) IIR filtre

2- WIENER FİLTRE TEORİSİ

2.1. Lineer Wiener Filtresi

Birim darbe cevabı h(1), h(2),..., h(M) ile verilen lineer bir FIR filtreyi göz önüne alalım. Şekil 2.1'de gösterildiği gibi filtre bir dizi gecikme elemanlarıyla (herbiri birim geciktirme elemanı Z^{-1} ile gösterilmiş), gerekli girişlere tekabül eden birim darbe cevabı katsayıları ve bir dizi toplama devresinden ibarettir. Filtrenin girişleri u(n), u(n-1),..., u(n-M+1) şeklinde M adet, istatistiksel anlamda rasgele (random) değişkenler dizisinden oluşmuş durumdadır. Bu dizi elemanlarının yine istatistiksel anlamda durağan (stationary) olduğunu kabul edeceğiz. Dizi durağan olduğuna göre aşağıdaki iki şartı sağlayacaktır.

1- Bu rasgele işlemenin (random process) ortalama değeri m ile gösterilirse, bunun n zaman değişkeninden bağımsız olması gerekir. Yani:

$$m \overset{\Delta}{=} E\{u(n)\} = \text{sabit} \qquad (2.1)$$

Burada E operatörü u(n) rasgele değişkeninin beklenen değerini göstermektedir.

2- Rasgele işlemenin özilişki fonksiyonu

$$r(n,m) \overset{\Delta}{=} E\{u(n)\, u(m)\} \qquad (2.2)$$

ile tarif edilir ve bu değer n-m zaman farkına bağlıdır. Yani:

$$r(n,m) = r(n-m) \qquad (2.3)$$

Filtre çıkışını y(n) ile gösterelim. Bunu aşağıdaki konvolusyonel toplam ifadesiyle gösterebiliriz:

$$y(n) = \sum_{k=1}^{M} h(k)\, u(n-k+1) \qquad (2.4)$$

Burada maksadımız arzu edilen bir d(n) filtre çıkışı ile, filtrenin reel çıkışı ya da cevabı olan y(n) arasındaki farkı birtakım yollarla minimize etmektir. Bu farkı şöylece gösterebiliriz:

$$e(n) = d(n) - y(n) \qquad (2.5)$$

Burada e(n)'e hata işareti (error signal) ya da kalan (residual) denir. Hata işaretinin sıfırdan farklı olması halinde filtre çıkışı y(n) ile arzu edilen d(n) cevabı aynı olmayacaktır. Bundan dolayı yapacağımız iş e(n) hatasını mümkün olduğu kadar küçük tutmaktır. İşte Wiener Teorisinde filtre, bu e(n) hata işaretinin ortalama karasel değeri minimum yapılarak optimize edilir.

2.2. Normal Denklemleri

Hata işaretinin ortalama karesel değerini

$$\varepsilon = E\{e^2(n)\} \qquad (2.6)$$

ile gösterilir. Bu ortalama karesel değer, reel ve pozitif bir skalar büyüklük olup, e(n) hata işaretinin 1 ohm'luk bir yük uçlarına uygulanmasıyla harcanan ortalama güce

eşittir. (2.5) ifadesini (2.6)'da yerine koyduğumuzda

$$\varepsilon = E\{d^2(n)\} - 2\,E\{d(n)\,y(n)\} + E\{y^2(n)\} \quad (2.7)$$

elde edilir. Giriş işareti u(n) ile, arzu edilen d(n) cevabının birleşik durağan (Jointly stationary) olduklarını kabul edersek, (2.7) ifadesinin sağ tarafı şöyle yorumlanabilir.

1- $E\{d^2(n)\}$ terimi, d(n)'in ortalama karesel değerine eşittir ve şöylece gösterilebilir.

$$P_d = E\{d^2(n)\} \quad (2.8)$$

2- (2.7) ifadesinde y(n) yerine (2.4) ile verilen değeri konulduğunda sırayla $E\{d(n)\cdot u(n-k+1)\}$ ve $E\{u(n-k+1)\,u(n-m+1)\}$ terimleri elde edilir. Bunlardan ilki olan, $E\{d(n)\,u(n-k+1)\}$ beklenen değeri, arzu edilen d'(n) işaretiyle, giriş işareti u(n)'in n = k-1'deki değeri olan u(n-k+1) arasındaki geçiş ilişkisi (cross-corelation) fonksiyonudur. Bunu da

$$P(k-1) = E\{d(n)\,u(n-k+1)\}, k = 1,2,\ldots, M \quad (2.9)$$

şeklinde ifade etmek mümkündür.

3- Son olarak, diğer beklenen değer olan $E\{u(n-k+1)\,u(n-m+1)\}$ değeri ise u(n) giriş işaretinin m-k için özilişki fonksiyonu olup

$$r(m-k) = E\{u(n-k+1)\,u(n-m+1)\} \quad (2.10)$$

ile gösterilebilir. Böylece (2.8), (2.9) ve (2.10)'daki kısaltmalar (2.7)'de yerine konursa ortalama karesel hata

işareti için

$$\varepsilon = P_d - 2 + \sum_{k=1}^{M} h(k)\, p(k-1) + \sum_{m=1}^{M} \sum_{k=1}^{M} h(k)h(m)r(m-k) \qquad (2.11)$$

ifadesi elde edilir. Ortalama karasel hatayı minimize etmek için, h(k)'ya göre sırasıyla k= 1,2,..., M için türevleri alınırsa

$$\frac{\partial \varepsilon}{\partial h(k)} = -2\, p(k-1) + 2 \sum_{m=1}^{M} h(m)\, r(m-k) \qquad (2.12)$$

ifadesi bulunur. Bu ifade sıfıra eşitlendiği taktirde filtre katsayılarının $h_o(1)$, $h_o(2)$,..., $h_o(M)$ biçiminde gösterilen optimum değerleri bulunur. Öyleki, bu optimum değerler kullanıldığı takdirde filtrenin hata çıkışı minimum olacaktır. Bu işlem yapılırsa aşağıdaki denklem takımını elde ederiz.

$$\sum_{m=1}^{M} h_o(m)\, r(m-k) = p(k-1), \quad k= 1,2,..., M \qquad (2.13)$$

Son elde ettiğimiz bu denklem takımına normal denklemleri denir. Neden bu ismin verildiği, bölüm 2.5'te açıklamacaktır. Bu denklem sisteminde özilişki katsayıları olan r(m-k) ile, geçiş-ilişki katsayıları olan p(k-1)'in değerleri bilinmektedir. Bilinmeyenler $h_o(1)$,..., $h_o(M)$ filtre katsayılarıdır.

2.3. Minimum Ortalama Karesel Hata

Ortalama karesel hatanın minimum değerini ε_{min} ile gösterelim. Minimum hata için, (2.11) ifadesinde p(k-1) yerine (2.13)'de elde edilen değeri yazılırsa

$$\varepsilon_{min} = P_d - 2 \sum_{k=1}^{M} h_o(k)\, P(k-1) + \sum_{k=1}^{M} \sum_{m=1}^{M} h_o(k)\, h_o(m)\, r(m-k)$$

$$= P_d - 2 \sum_{k=1}^{M} h_o(k) \{2P(k-1) - \sum_{m=1}^{M} h_o(m)\, r(m-k)\}$$

$$= P_d - \sum_{k=1}^{M} h_o(k)\, P(k-1) \qquad (2.14)$$

ifadesi bulunur. İste bütün bu minimizasyon işlemleri yapıldığı takdirde ortalama karesel anlamda optimum filtre elde edilerek (2.14) ile verilen minimum hataya erişilir.

2.4. Ortogonallik Prensibi

(2.15) ile verilen normal denklemleri, minimum ortalama karesel hata anlamında optimum filtre katsayılarını tarif eder. P(k-1) ve r(k-m)'nin tanımlarını kullanarak bu ifadeyi tekrar yazabiliriz.

$$\sum_{m=1}^{M} h_o(m)\, E\{u(n-m+1)\, u(n-k+1)\} = E\{d(n)\, u(n-k+1)\}$$

$$k = 1,2,\ldots,M \qquad (2.16)$$

Bu ifadeyi düzenlersek;

$$E\left[\{d(n) - \sum_{m=1}^{M} h_o(m)\, u(n-m+1)\}\, u(n-k+1)\right] = 0$$

$$k = 1,2,\ldots, M \qquad (2.17)$$

(2.17) ifadesinde eşitliğin sol tarafındaki toplam ifadesi filtrenin optimum çıkış değeri olan $y_o(n)$'i verecektir.

$$y_o(n) = \sum_{m=1}^{M} h_o(m)\, u(n-m+1) \qquad (2.18)$$

istenen filtre cevabı d(n) ile $y_o(n)$ arasındaki fark ise minimum ya da optimum hatayı vereceğinden (2.17) ifadesini tekrar şu şekilde yazabiliriz.

$$E\ \{d(n) - y_o(n)\}\ u(n-k+1) = E\ \{e_o(n)\ u(n-k+1)\}$$
$$= 0, \quad k=1,2,\ldots, M \qquad (2.19)$$

Burda $e_o(n)$ optimum filtre çıkışını sağlayan minimum hata işaretidir. (2.19) ifadesinden görüldüğü gibi optimum filtre için $e_o(n)$ hata işaretiyle filtre girişleri birbirine ortogonaldirler. Buna ortogonallik prensibi denir. Ortogonallik prensibinin bir neticesi olarak $e_o(n)$ hata işaretiyle optimum filtre çıkışı $y_o(n)$'in de birbirlerine ortogonal olduklarını söyleyebiliriz.

$$E\ \{e_o(n)\ y_o(n)\} = 0 \qquad (2.20)$$

Bunun nasıl olduğunu ise şöylece gösterebiliriz.

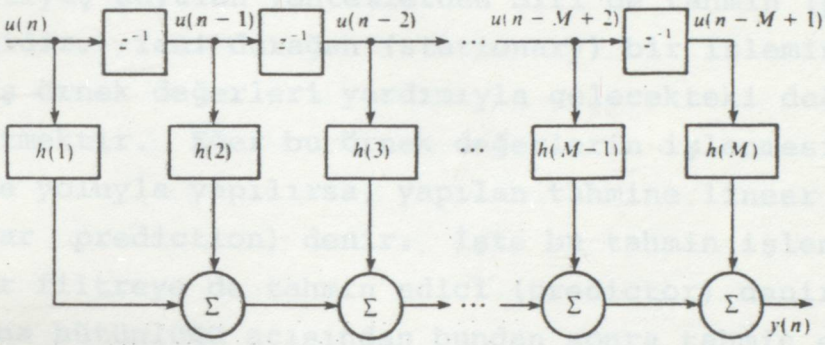
$$E\ \{e_o(n)\ y_o(n)\} = E\ \{e_o(n) \sum_{k=1}^{M} h_o(k)\, u(n-k+1)\}$$

$$= \sum_{k=1}^{M} h_o(k)\ \{E\ e_o(n)\ u(n-k+1)\}$$

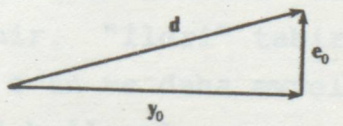
$$= 0$$

(2.20) denklemi ilginç bir geometrik yoruma sahiptir. Şayet filtre çıkışını, arzu edilen cevabı ve hata işaretini gösteren rasgele (random) değişkenleri birer vektör olarak gösterirsek arzu edilen filtre cevabı, filtre çıkışı ile hata işaretinin vektörel toplamına eşit olacaktır. Bu üç

büyüklük arasındaki ilişki Şekil 2.2'de geometrik olarak verilmiştir. Şekilden görüleceği gibi $e_o(n)$ hata vektörüyle $y_o(n)$ optimum çıkış vektörü birbirine diktir. Zaten bu diklik ya da normallik yüzünden "normal denklemleri" tabiri kullanılmıştır.



Şekil 2.1: Birim darbe cevabı katsayıları cinsinden FIR filtre



Şekil 2.2:
Ortogonallik prensibinin geometrik gösterimi

3- LİNEER TAHMİN (LİNEAR PREDİCTION)

İşaret işlemede (signal processing) çok kullanılan ve ihtiyaç duyulan yöntemlerden biri de tahmin (pediction) işlemidir. Yani durağan (stationary) bir işlemin, bir grup geçmiş örnek değerleri yardımıyla gelecekteki değerini tahmin etmektir. Eğer bu örnek değerlerin işlenmesi lineer filtreleme yoluyla yapılırsa, yapılan tahmine lineer tahmin (linear prediction) denir. İşte bu tahmin işlemini yapan lineer filtreye de tahmin edici (predictor) denir. Kullanım ve mana bütünlüğü açısından bundan sonra tahmin edici yerine orjinal ismi olan "predictör" ismini kullanacağız. Rasgele işlemenin (ramdom process) belli bir andaki örneğine ait arzu edilen değerle prediktörün reel çıkışı arasındaki farka tahmin hatası (prediction error) denir. Wiener filtre teorisine göre predictor, bu tahmin hatasının karesel ortalamasının minimize edilmesiyle optimize edilip dizayn edilir. Geçen bölümde filtreleri anlatırken Wiener filtresine de değinilmişti.

Yukarda anlatılan tahmine "ileri tahmin" (forward prediction) da denir. "İleri" tabiri, filtrenin belli bir andaki çıkışının, o an ve daha evvelki girişlerden tahmin edilmesinden ötürü kullanılmıştır. Yani filtre geçmişe bakarak ileri doğru gitmektedir. Bundan başka geri tahmin (backward prediction) diyebileceğimiz bir tahmin daha mümkündür ki, bu durumda filtre belli bir anda ileriye bakarak gerideki bir değeri tahmin eder.

Wiener filtre teorisinin lineer tahmin (linear prediction) problemine uygulanmasıyla karşımıza Şekil 2.1'dekine benzer bir filtre yapısı çıkar. Ancak bu yapı ileri

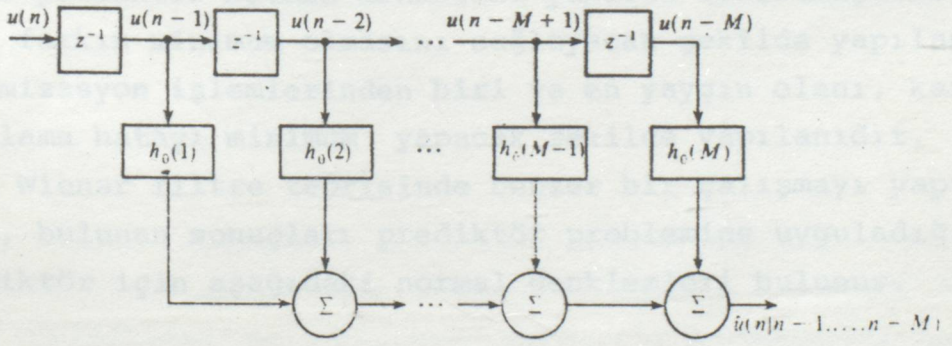
ve geri tahminler için ayrı ayrı mümkündür. İleri ve geri tahmin filtre yapıları birleştirildiğinde latis tahmin (lattice predictor) yapısı karşımıza çıkar. Latis tahmin filtresi işaret işleme açısından ilginç ve kullanışlı bir yapıya sahiptir.

3.1. İleri Lineer Tahmin Normal Denklemleri

Durağan bir işaretten elde edilen u(n-1), u(n-2),..., u(n-M) zaman dizisini ele alalım. İleri lineer tahmin (forward linear prediction) probleminde bu örnek değerler u(n) değerini tahmin etmek için kullanılır. u(n)'in bu notasyonu hernekadar biraz sıkıcı gibi görünüyorsa da olayın anlaşılması için uygundur. Buna göre ileri tahmin hatasını şöylece yazabiliriz:

$$f_M(n) = u(n) - \hat{u}(n/n-1,\ldots,n-M) \qquad (3.1)$$

Burada u(n) arzu edilen cevaptır. $f_M(n)$'deki M indisi tahmin işleminin M adet geçmiş örnek yardımıyla yapıldığını gösterir. Böyle bir prediktörün yapısı, şematik olarak Şekil 3.1'de gösterilmiştir. Şimdi buradaki problem (3.1)



Şekil 3.1: İleri lineer tahmin filtresi

bağıntısıyla verilen prediktör hatasının karesel ortalamasını minimum yapacak şekilde bu filtrenin optimize edilmesidir.

Bu optimizasyon işlemindeki anahtar parametrelerin filtre katsayıları olduğunu 2. bölümde görmüştük. Bölüm 2'de Wiener filtre teorisini anlatırken elde ettiğimiz sonuçları, gerekli değişiklikleri yaparak lineer prediktör problemine uygulayabiliriz. Şekil 3.1'deki prediktör filtresine göre aşağıdaki bağıntıyı yazabiliriz.

$$\hat{u}(n/n-1,\ldots,n-M) = \sum_{k=1}^{M} h_o(k)\, u(n-k) \qquad (3.2)$$

Burada u(n-1), u(n-2),..., u(n-M) gerçek değerini bulmak istediğimiz u(n) dizisinin geçmiş elemanlarıdır. Şekil 3.1'e tekrar dikkatlice baktığımızda buradaki ana temanın şu olduğu görülür. Prediktör girişine bir u(n) dizisi girmektedir. Prediktör adından anlaşılacağı üzere dizinin n. örneğinin değerini daha evvel ölçmüş olduğu geçmiş örnek değerlerden tahmini olarak hesaplar ve olay her adım için böyle tekrar eder. Tahmin edilen herbir andaki örnek, bir sonraki an için bilinen bir örnek kabul edilerek tahmin işlemine devam edilir. Filtre katsayıları öyle seçilmelidir ki, tahmin edilen değer ile gerçek değer arasındaki fark minimum olsun. Bu fark (3.1) ifadesi ile verilmiş olup buna ileri prediktör hatası dendiğini yukarda belirtmiştik. Aradaki farkın minimum olmasını sağlayacak şekilde yapılan optimizasyon işlemlerinden biri ve en yaygın olanı, karesel ortalama hatayı minimum yapacak şekilde yapılanıdır. Bölüm 2'de Wienar filtre teorisinde benzer bir çalışmayı yaptığımız için, bulunan sonuçları prediktör problemine uyguladığımızda, prediktör için aşağıdaki normal denklemleri bulunur.

$$\sum_{m=1}^{M} h_o(m)\, r(m-k) = r(k), \quad k=1,2,\ldots M \qquad (3.3)$$

Burada r(m-k) girişin geçmiş değerleri arasındaki ilişki

fonksiyonu olup şu bağıntı ile verilir:

$$r(m-k) = E\{u(n-k)\, u(n-m)\}, \quad k=1,2,\ldots, M \qquad (3.4)$$

r(k) ise arzu edilen u(n) ile u(n)'in geçmişteki değerleri arasındaki ilişki fonksiyonu olup, genelde özilişki fonksiyonu denir.

$$r(k) = E\{u(n)\, u(n-k)\}, \quad k= 1,2,\ldots, M$$

(3.3) bağıntısından görüldüğü gibi optimum prediktöre ulaşmak için özilişki fonksiyonu katsayılarını bilmek kâfidir.

Yine daha önce yaptığımız incelemeye benzer olarak ileri prediktör hatası için karesel ortalama değeri şöyle ifade edebiliriz:

$$P_{f,M} = E\{f_M^2(n)\}$$

$$= r(o) - \sum_{m=1}^{M} h_o(m)\, r(m) \qquad (3.5)$$

Burada r(o), arzu edilen cevap olan u(n)'in karesel ortalama değeri olup,

$$r(o) = E\{u^2(n)\}$$

bağıntısıyla verilir. (3.3)'deki normal denklemleri ile (3.5)'deki ortalama hata ifadesini, $h_o(1),\ldots, h_o(M)$ filtre katsayılarından yeni bir grup prediktör katsayıları tanımlayarak birleştirmek mümkündür. Prediktör katsayılarının tanımı aşağıdaki gibidir.

$$a_M(n) = \begin{cases} 1 & , \quad m=0 \\ -h_o(m) & , \quad m=1,\ldots,M \\ 0 & , \quad m>M \end{cases} \qquad (3.6)$$

Tanımladığımız bu prediktör katsayıları yardımıyla şimdi birleştirme işlemini yapabiliriz. Bunun için (3.3) ifadesinde r(k)'yı eşitliğin sol tarafındaki toplama işlemine, (3.5) ifadesindeki r(o) teriminide yine toplama işlemine dahil etmek kafidir. Yani (3.3) ve (3.5) ifadeleri yerine sırasıyla

$$\sum_{m=o}^{M} a_M(m)\, r(m-k) = 0, \quad k=1,2,\ldots, M \qquad (3.7)$$

$$\sum_{m=o}^{M} a_M(m)\, r(m) = P_{f,M} \qquad (3.8)$$

bağıntılarını yazmak mümkündür. Bu iki ifade şimdi dikkat edilirse $a_M(n)$ katsayıları yardımıyla

$$\sum_{m=o}^{M} a_M(m)\, r(m-k) = \begin{cases} P_{f,M} & , \quad k=o \\ 0 & , \quad k=1,2,\ldots, M \end{cases} \qquad (3.9)$$

şeklinde birleştirilebilir. Elde ettiğimiz bu son ifadeye ileri prediktör filtresi için "Genelleştirilmiş Normal Denklemleri" denir.

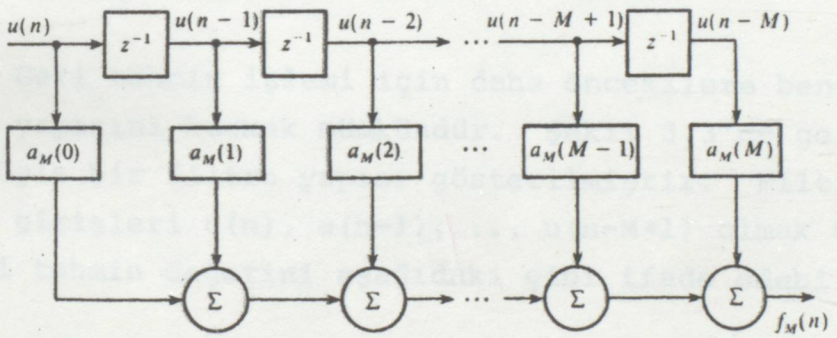
İleri prediktör hatası için de prediktör katsayıları cinsinden yeni bir ifade bulmak mümkündür. Bunu aşağıdaki gibi yazabiliriz:

$$f_M(n) = u(n) - \hat{u}(n/n-1,\ldots,n-M)$$

$$= u(n) - \sum_{k=1}^{M} h_o(k)\, u(n-k)$$

$$= \sum_{k=0}^{M} a_M(k)\, u\,(n-k) \qquad (3.10)$$

(3.10) ifadesi "Prediktör Hata Filtresi" diyebileceğimiz yeni bir filtreyi tarif eder. Bu yapı şematik olarak Şekil 3.2'de gösterilmiştir. Görüldüğü gibi bu hata filtresi daha önce tarif edilen prediktör filtresinden bir adım daha uzundur. Prediktör filtresi M adımlı, prediktör hata filtresi ise M+1 adımlıdır.



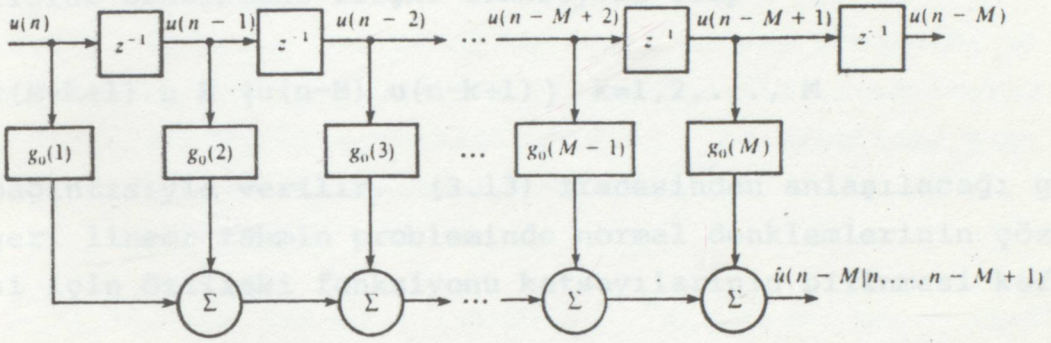
Şekil 3.2 : İleri tahmin-hata filtresi.

3.2. Geri Lineer Tahmin İçin Normal Denklemleri

Şimdi de geri lineer tahmin (backward linear prediction) problemini ele alalım. Bu durumda u(n), u(n-1),..., u(n-m+1) dizisi, u(n-M) değerini tahmin etmek için kullanılır. Bu tahminin neticesini u(n-M/n,..., n-m+1) ile gösterelim. u(n-M)'de dizinin n-M anındaki gerçek değerini göstermek üzere geri lineer tahmin hatasını

$$b_M(n) = u(n-M) - \hat{u}(n-M/n,\ldots, n-m+1) \qquad (3.11)$$

şeklinde ifade edebiliriz. Burada u(n-M) arzu edilen cevap rolündedir. $b_M(n)$ terimindeki M indisi, tahmin işleminde M adet örneğin kullanıldığını belirtir.



Şekil 3.3 : Geri Lineer tahmin filtresi.

Geri tahmin işlemi için daha öncekilere benzer bir filtre yapısını kurmak mümkündür. Şekil 3.3'de geri tahmin için böyle bir filtre yapısı gösterilmiştir. Filtrenin ana ve yan girişleri u(n), u(n-1),..., u(n-M+1) olmak üzere çıkışındaki tahmin değerini aşağıdaki gibi ifade edebiliriz.

$$\hat{u}(n-M/n,\ldots,n-M+1) = \sum_{k=1}^{M} g_o(k)\, u(n-k+1) \qquad (3.12)$$

Burada $g_o(1)$, $g_o(2)$,..., $g_o(M)$ karesel ortalama anlamında optimize edilmiş prediktör katsayılarıdır. Optimizasyon işlemi tahmin edileceği gibi yine, normal denklemlerden hareketle yapılır. Daha evvel elde edilen sonuçları kullanarak geri lineer tahmin için normal denklemlerini

$$\sum_{m=o}^{M} g_o(m)\, r(m-k) = r(M-k+1), \quad k=1,2,\ldots, M \qquad (3.13)$$

olarak yazabiliriz. Burada r(m-k) girişler arasındaki ilişkiyi gösteren ilişki fonksiyonu olup

$$r(m-k) = E\{u(n-k+1)\, u(n-m+1)\}, \quad k=1,2,\ldots, M \qquad (3.14)$$

bağıntısıyla verilir. r(M-k+1) ise artış edilen cevapla girişler arasındaki ilişki fonksiyonu olup

$$r(M-k+1) = E\{u(n-M)\, u(n-k+1)\} \quad k=1,2,\ldots, M$$

bağıntısıyla verilir. (3.13) ifadesinden anlaşılacağı gibi geri lineer tahmin probleminde normal denklemlerinin çözülmesi için özilişki fonksiyonu katsayılarının bilinmesi kafidir.

Geri lineer tahmin preblemi için karesel ortalama hata aşağıdaki bağıntıyla verilir:

$$P_{b,M} = E\{b_M^2(n)\}$$

$$= r(o) - \sum_{m=1}^{M} g_o(m)\, r(M-m+1) \qquad (3.15)$$

Burada r(o) arzu edilen cevabın karesel ortalama değeridir ve r(M-k+1) ise arzu edilen cevabla girişler arasındaki ilişkidir. (3.13) ifadesinde m yerine M-m+1 ve k yerine M-k+1 dönüşümleri yaptığımızda karşımıza ilginç bir sonuç çıkar.

$$m = M-m+1$$
$$k = M-k+1$$

$$\sum_{m=1}^{M} g_o(M-m+1).r(m-k) = r(k), \quad k=1,2,\ldots, M \qquad (3.16)$$

En son elde ettiğimiz bu (3.16) ifadesiyle daha önce ileri lineer tahmin için elde ettiğimiz normal denklemlerini birebir karşılaştırdığımızda şunlar elde edilir:

$$h_o(m) = g_o(M-m+1), \quad m=1,2,\ldots, M \tag{3.17}$$

$$g_o(m) = h_o(M-m+1), \quad m=1,2,\ldots, M \tag{3 18}$$

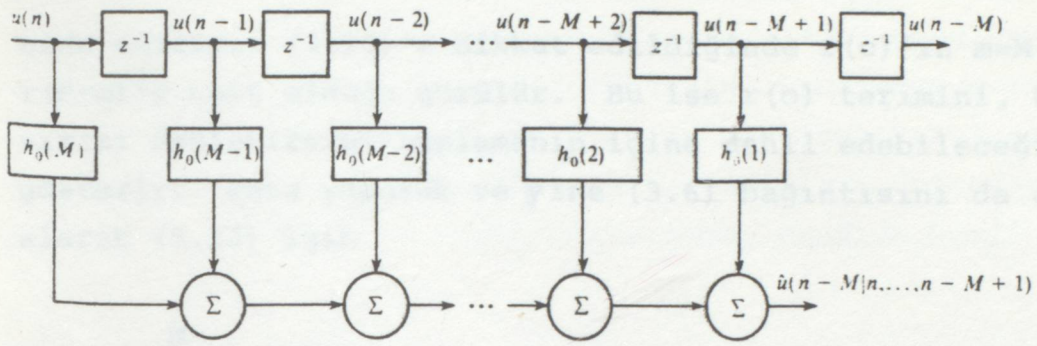
(3.18) ifadesine baktığımızda, geri lineer prediktör filtresini ileri lineer prediktör filtresine ait katsayılarla yeniden düzenleyebileceğimizi görürüz. Bu düzenleme yapıldığında (3.13)'deki normal denklemlerini

$$\sum_{m=1}^{M} h_o(M-m+1)\, r(m-k) = r(M-k+1), \quad k=1,2,\ldots M \tag{3.19}$$

şeklinde ileri lineer prediktör filtre katsayıları cinsinden buluruz. (Bak. Şekil 3.5) Yukarda yaptığımız m=M-m+1 dönüşümünü (3.15) ifadesi içinde yaptığımızda başka bir ilginç sonuç daha karşımıza çıkar:

$$\begin{aligned} P_{b,M} &= r(o) - \sum_{m=1}^{M} g_o(m)\, r(M-m+1) \\ &= r(o) - \sum_{m=1}^{M} g_o(m-m+1)\, r(m) \\ &= r(o) - \sum_{m=1}^{M} h_o(m)\, r(m) \\ &= P_{f,M} \end{aligned} \tag{3.20}$$

(3.20) ifadesinden görüleceği gibi durağan bir işlem için (stationary process) ileri tahmin hatası $f_M(n)$ ile, geri tahmin hatası $b_M(n)$ karesel ortalama anlamında aynı değere sahiptirler.



Şekil 3.4 : Geri lineer tahmin filtresinin, ileri lineer tahmin filtre katsayıları cinsinden gerçekleştirilmesi.

Tekrar (3.19) ifadesine dönerek incelememize devam edelim. Bu ifadede m=m-1, k=k-1 dönüşümlerini yapalım. Bunu yaptığımızda (3.19) ifadesi yerine

$$\sum_{m=o}^{M-1} h_o(M-m)\, r(m-k) = r(M-k), \quad k=o,1,\ldots, M-1 \qquad (3.21)$$

bağıntısını buluruz. Şimdi bu ifadenin sağ tarafındaki terimi (3.6) bağıntısınıda dikkate alarak toplamanın içine dahil edebiliriz. Gerçekten de bunu yaptığımızda geri lineer prediktör filtresine ait normal denklemlerini, ileri lineer prediktör katsayıları cinsinden değişik bir formda elde ederiz. Yani:

$$\sum_{m=o}^{M} a_M(M-m)\, r(m-k) = 0 \quad k= 0,1,\ldots, M-1 \qquad (3.22)$$

Benzer şekilde geri lineer prediktör filtresine ait karesel ortalama hata ifadesini de yine $h_o(m)$ katsayıları cinsinden yazabiliriz. Bu yapıldığında:

$$P_{b,M} = r(o) - \sum_{m=o}^{M-1} h_o(M-m)\, r(M-m) \qquad (3.23)$$

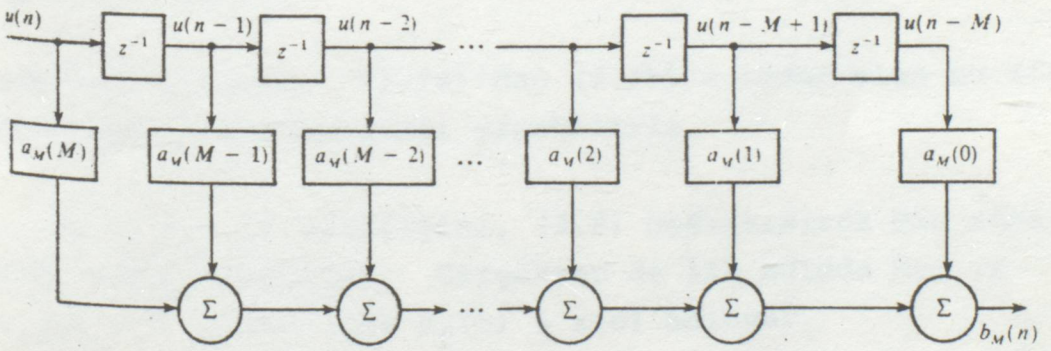
elde edilir. (3.23)'e dikkat edildiğinde r(o)'ın m=M için r(M-m)'e eşit olduğu görülür. Bu ise r(o) terimini, üst sınırını değiştirerek toplamanın içine dahil edebileceğimizi gösterir. Bunu yaparak ve yine (3.6) bağıntısını da dikkate alarak (3.23) için

$$P_{b,M} = \sum_{m=o}^{M} a_M(M-m).r(m-M) \qquad (3.24)$$

eşitliğini buluruz. (3.24) eşitliğine baktığımızda, k=M için (3.22)'nin (3.24)'e eşit olduğu görülür. Bu da bize bu iki bağıntıyı tek bir ifade altında birleştirebileceğimizi ima eder. Gerçekten bunu yaptığımızda

$$\sum_{m=}^{M} a_M(M-m)\ r(m-k) = \begin{cases} 0 & , \ k = 0,1,\ldots M \\ P_{b,M} & , \ k = M \end{cases} \qquad (3.25)$$

ifadesini buluruz. En son elde ettiğimiz bu eşitliğe, geri lineer prediktör filtresi için "Genelleştirilmiş Normal Denklemleri" denir.



Şekil 3.5 : Geri tahmin-hata filtresi.

3.3. Levinson-Durbin Algoritması

Burada ileri lineer prediktör filtresine ait normal

denklemlerin çözümünü rekürsif bir tarzda veren Levinson-Durbin algoritmasını sunacağız. Ancak sadece sonuçlar verilecek olup, bu sonuca nasıl gelindiği incelenmeyecektir. İşlenen u(n) dizisinin deterministik olduğunu kabul ederek beklenen değerler yerine gerekli karşılıkları doğrudan yazılmıştır. Buna göre çözüm algoritması aşağıda verilmiştir.

$$P_f(o) = r(o)$$

$$k(i) = -\{r(i) \sum_{j=1}^{i-1} a(j,i-1)\, r(i-j)\}/E(i-1) \qquad (3.26)$$

$$a(i,i) = k(i)$$

$$a(j,i) = a(j,i-1) \quad k(i)\, a(i-j,i-1),\ 1 \leq j \leq i-1 \qquad (3.27)$$

$$P_f(i) = (1-k^2(i))\, P_f(i-1) \qquad (3.28)$$

yukardaki denklemler rekürsif bir tarzda $1 \leq i \leq M$ için çözüldükten sonra, son olarak prediktör katsayıları

$$a_M(i) = a(i,M), \qquad , \quad 1 \leq i \leq M \qquad (3.29)$$

şeklinde bulunur. (3.26)'dan (3.29)'a kadar olan bu ifadelerden şu açıklamaları yapabiliriz.

1- $P_f(o) = r(o)$ eşitliğini, (3.8) bağıntısında M=o koyarak elde etmek mümkündür. Gerçekten de ilk adımda M=o ve $a_o(o) = 1$ olduğu için $P_f(o) = r(o)$ bulunur.

2- k(i) katsayıları literatürde yansıma katsayıları diye tabir edilir. Bu katsayılar her adım için

$$|k(i)| \leq 1$$
$$|k(i+1)| \leq |k(i)|$$

bağıntılarını sağlarlar. Bu katsayılar özellikle ses sentezinin yapılması halinde bir test parametresi olarak kullanılmaya çok elverişlidir. Bu yüzden önemi büyüktür.

3- Yukardaki bağıntılarda kullanılan özilişki katsayıları şu bağıntıyla verilir:

$$r(i) = \sum_{m=o}^{N-1-i} u(m)\, u(m-i) \qquad (3.30)$$

Burada N, üzerinde işlem yapılan dizinin uzunluğudur. Burada şu noktaya açıklık kazandırmak gerekir, şöyleki; şu ana kadar yapılan incelemelerde prediktör hata filtresinin derecesi olan M aynı zamanda sanki dizinin uzunluğu gibi anlaşılmış olabilir. Bu durum mümmündür. Ancak dizi uzunluğu N, M ile sınırlı değildir. N dizi uzunluğunu belirleyen faktörlerden en önemlisi, işaretin durağan olduğu süredir. Eğer işaret, sürekli durağan bir işaretse, analiz aralığı da diyebileceğimiz N dizi uzunluğu, istenen uygun bir değerde seçilebilir. Ancak işaretin kesikli durağan olması halinde analiz aralığı olarak, işaretin durağan olduğu, ortalama bir süreyi almak gerekir. Mesela ses işaretleri bu şekilde kesikli durağan bir yapıya sahiptir. Daha sonra da bahsedildiği gibi, ses işareti bir sesli harfin söylenişine aitse, belli bir pitch peryoduyla tekrar eden durağan bir yapıdadır. Ses işareti eğer bir sessiz harfin söylenişine aitse bu durumda işaretin yüksek frekans bileşenleri çok yüksek olacağından yine kendi içinde bir durağanlığa sahiptir. Dolayısıyla ses işaretinin incelenmesinde önce işaretin seslimi yoksa sessizmi harflere ait olduğunu yani durağanlık süresini belirlemek gerekir. Pratikte ses işaretleri için bu durağanlık süresi 10 ila 30 msn arasında değişir. Ses analizi

esnasında bu süreyi ya adaptif olarak belirlemek gerekir. Ya da ortalama bir değer almak gerekir. (3.30) ile verilen özilişki bağıntısındaki u(m) giriş dizisini esasen bir pencere fonksiyonu ile çarpmak gerekir. Bu, özilişki katsayısının tanımından gelen bir kısıtlamadır ve bunu şöylece gösterebiliriz:

Özilişki fonksiyonu en genelde

$$r(i) = \sum_{m=-\infty}^{\infty} u(m)\, u(m-i) \qquad (3.31)$$

bağıntısıyla verilir. Ancak yukarda açıkladığımız nedenlerden ötürü pratikte böyle sınırsız bir analiz aralığı almak mümkün değildir. Bu durumda işaretin durağan olduğu süre dikkate alınarak belli bir dizi uzunluğu seçmek gerekir. Yani işaret durağan olduğu süre içinde pencerelenerek incelenmelidir. W(n) ile göstereceğimiz ve

$$W(n) = \begin{cases} W(n) & 0 \leq n \leq N-1 \\ 0 & \text{dışında} \end{cases} \qquad (3.32)$$

şeklinde tanımlayabileceğimiz maksada uygun bir pencere forksiyonu ile giriş dizisini çarpmak gerekir. Bu durum dikkate alındığında özilişki fonksiyonu için;

$$r(i) = \sum_{m=o}^{N-1-i} u'(m)\, u'(m-i) \qquad (3.33)$$

yazmak mümkündür. Burada u'(m)

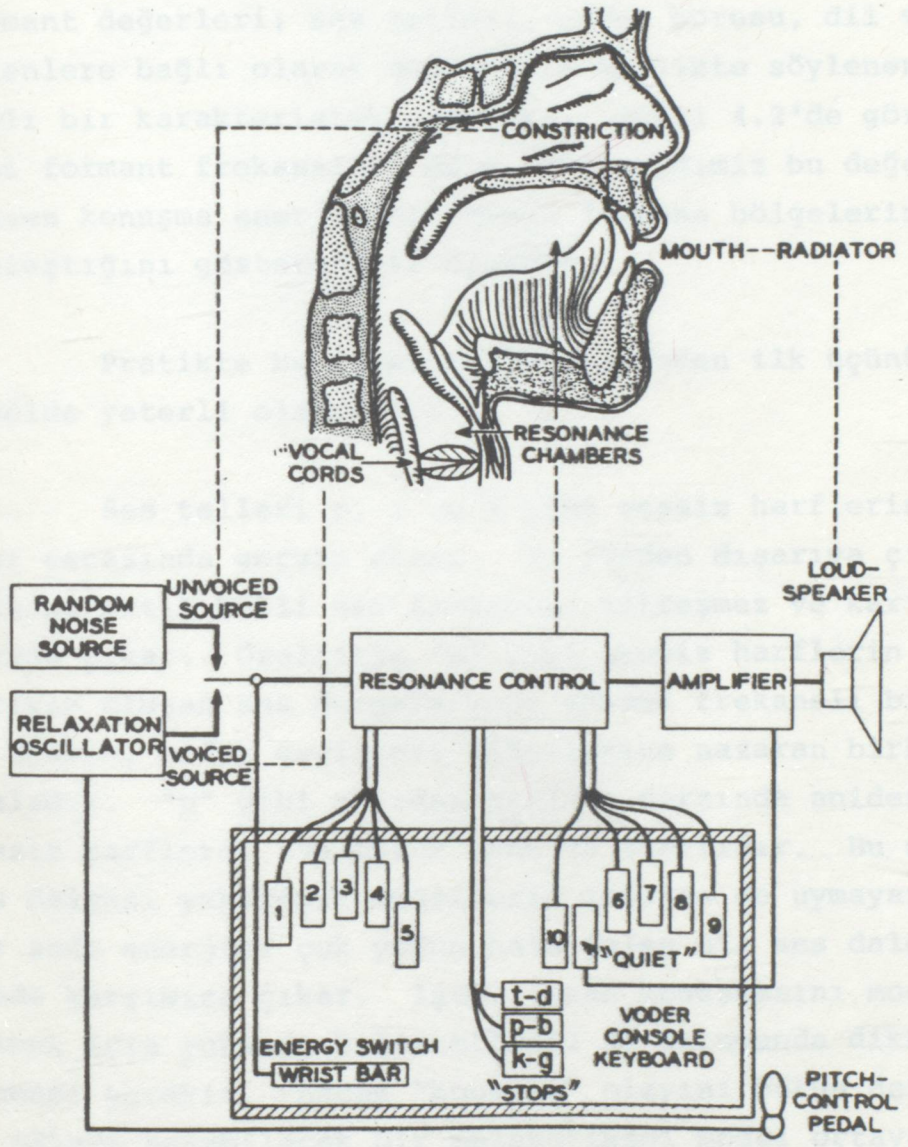
$$u'(m) = \begin{cases} u(m).W(n) & 0 \leq n \leq N-1 \\ 0 & \text{dışında} \end{cases} \qquad (3.24)$$

bağıntısıyla belirlidir.

4- KONUŞMA ÜRETİMİNİN MEKANİZMASI

Şekil 4.1'de insan konuşmasının üretim mekanizması şematik olarak gösterilmiştir. Normal ses üretiminde, konuşmadan hemen önce otomatik olarak göğüs genişler, ciğerler havayla dolar. Bu hava konuşma esnasında nefes yollarından geçerek dışarı itilir. Havanın bu dışarı itilmesi esnasında sesli harfleri söylerken olduğu gibi ses telleri gerginse, ses yolu titreşim ya da osilasyon modunda çalışır. Gerçekten de, herhangibir sesli harfi söylerken sesi çıkarmadan hemen önce ses telleri gergin hale gelir. Konuşma anında dışarı itilen havanın zorlamasıyla bu teller belli bir frekansta titreşir. Dolayısıyla dışarı çıkan ve algılanan ses dalgası aynı frekansta titreşen kısa süreli bir dalga olacaktır. Ses tellerinin bu titreşim frekansının tersi olan titreşim peryoduna "Pitch Peryodu" denir. Pitch Peryodu, sesli harflerin tipik bir özelliği olup, insandan insana çok az farkeden bir karakteristik değerdir. Bu yüzden ses analiz ve sentez sitemlerinde, özellikle yapay ses üretiminde kullanılan temel öğelerden biridir. Ses tanıma sistemlerinde de yine, ayıraç olarak kullanılabilen bir karakteristiktir. Pitch peryodunun tipik değeri 10 ile 20 ms arasında değişir.

Ses yolu (Vocal Tract) bir bütün olarak ciğer boşluklarından dudaklara dek uzanan üniform olmayan akustik bir tüp gibidir. Yetişkin bir bayanda bu borunun uzunluğu tipik olarak 17 cm civarındadır. Bu nonüniform tüpün kesiti, tüp boyunca yayılan diğer organlarında etkisiyle 0 cm^2 den 20 cm^2 ye kadar değişen bir değere sahiptir. Ses

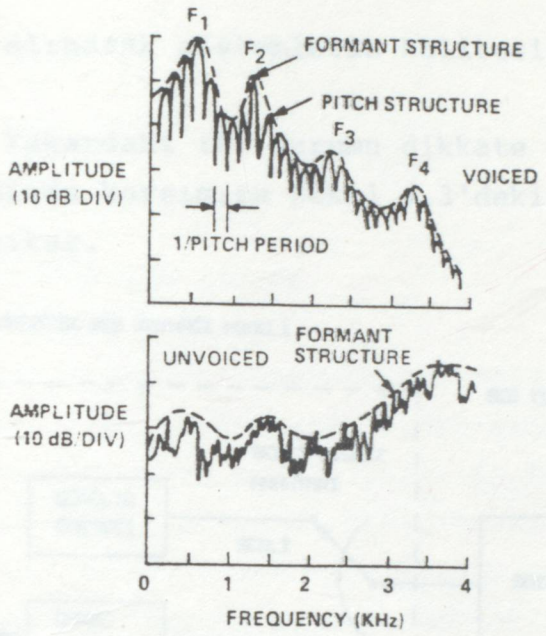


Şekil 4.1: İnsanda ses üretim mekanizması

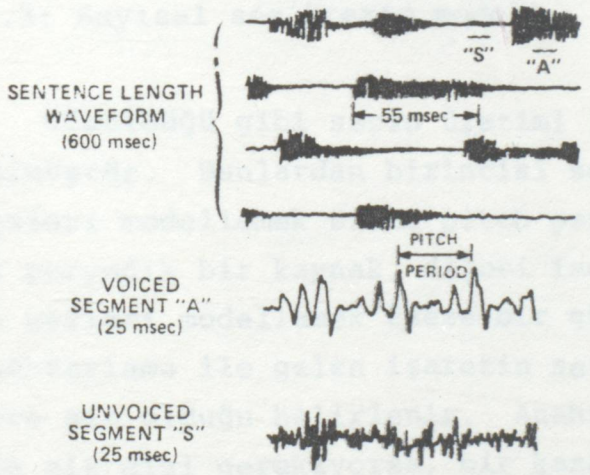
yolu akustik bir tüp olması nedeniyle "Formantlar" adını verdiğimiz bazı rezonans modlarında titreşim yapar. Bu formant değerleri; ses telleri, nefes borusu, dil v.s gibi etkenlere bağlı olarak değişmekle birlikte söylenen söze bağlı bir karakteristik değerdir. Şekil 4.2'de görüleceği gibi formant frekansları diye belirttiğimiz bu değerler, esasen konuşma enerjisinin hangi frekans bölgelerinde yoğunlaştığını gösteren bir ölçüdür.

Pratikte bu formant değerlerinden ilk üçünü almak genelde yeterli olmaktadır.

Ses telleri p, z ve s gibi sessiz harflerin söylenmesi esnasında gergin olmaz. Bu yüzden dışarıya çıkan ses dalgası artık belli bir frekansta titreşmez ve karışık bir tarzda çıkar. Özellikle "s" gibi sessiz harflerin söylenmesiyle oluşan ses dalgalarının yüksek frekanslı bileşenleri baskın, sıfır geçişleri diğerlerine nazaran birkaç kat fazladır. "p" gibi ağızdan patlama tarzında aniden çıkan sessiz harflerde ise durum daha da farklıdır. Bu durumda ses dalgası yukardaki modellerin ikisine de uymayarak, belli bir anda enerjisi çok yoğun hale gelen bir ses dalgası şeklinde karşımıza çıkar. İşte, insan konuşmasını modelleyebilmek için yukarda belirttiğimiz üç durumunda dikkate alınması gerekir. Ancak "konuşma" olayını bütün detaylarıyla ortaya koyabilecek bir matematiksel model ortaya koymak hemen hemen imkansız gibidir. Çünkü "konuşma", konuşmacının psikolojik halinden ve konuşma ortamından tutunda, konuşmanın süreklimi yoksa kesiklimi olmasına bağlılığı gibi, dıştan müdahale edilemeyen bir çok nedenlere bağlıdır. Mamafih pratikte, yukarda anlatılan ilk iki tip "konuşma" çeşidini modellemek mümkün ve çoğu kez yeterli olur. "Konuşma" ile alakadar ses sentezi ve özellikle ses tanıma sistemlerinin tasarımlarında ihtiyaca göre sesin dikkate alınması gereken bir çok özelliği mevcuttur. Bunlar da



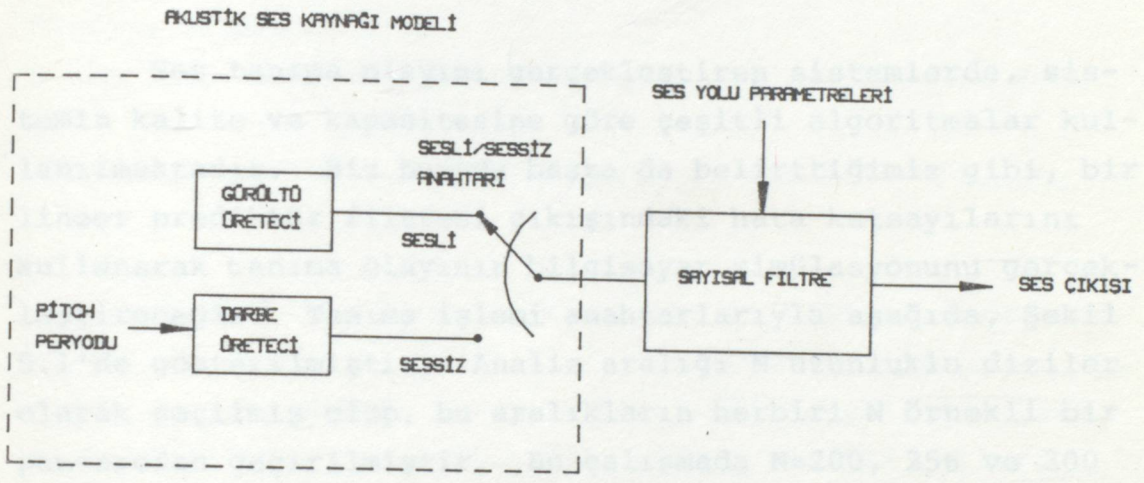
Şekil 4.2: (a) Sesli ve sessiz harflere ilişkin pitch ve formant yapısının frekansla değişimi.



Şekil 4.2: (b) Sesli ve sessiz harflere iliskin dalga şekli örnekleri

dikkate alınarak sistemlerin kalitesi artırılabilir.

Yukardaki iki durumu dikkate alarak bir modelleme yaptığımızda karşımıza Şekil 4.3'deki sayısal ses üretim modeli çıkar.



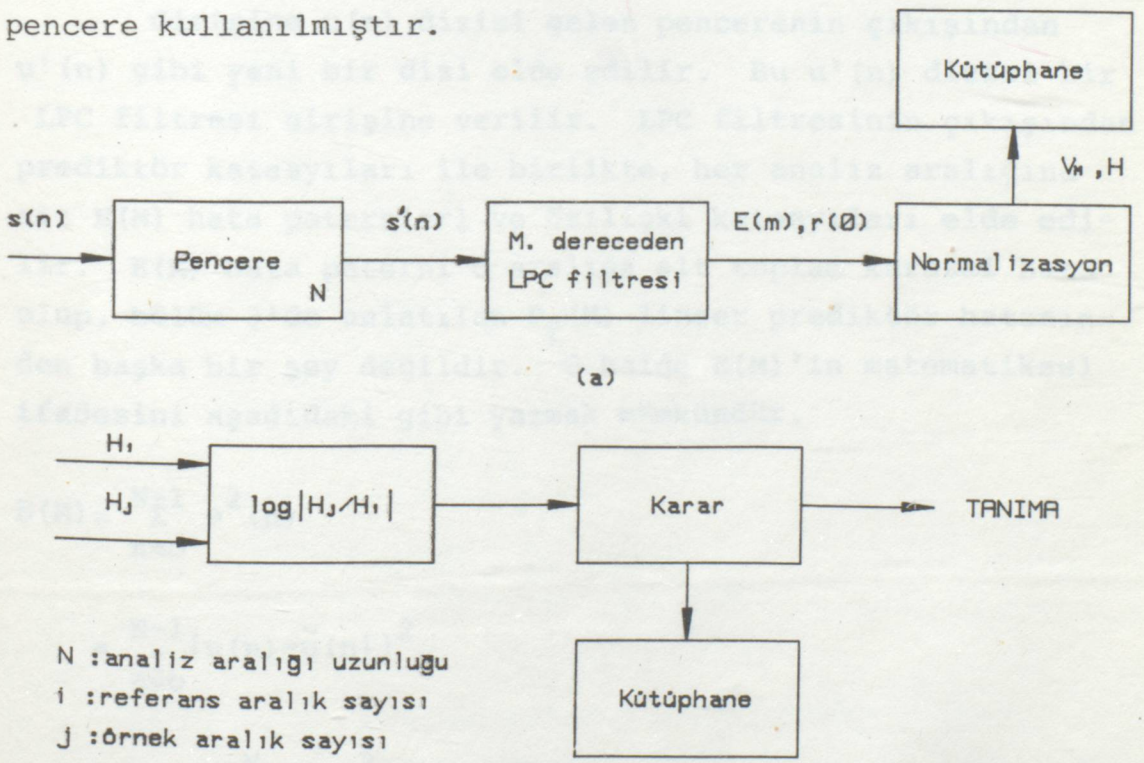
Şekil 4.3: Sayısal ses üretim modeli.

Görüldüğü gibi sesin üretimi için iki ana kaynak düşünülmüştür. Bunlardan birincisi sesli harflere ilişkin sesleri modellemek üzere pitch peryoduna sahip darbeler üreten peryodik bir kaynak, diğeri ise sessiz harflere ilişkin sesleri modellemek üzere bir gürültü kaynağıdır. Bir anahtarlama ile gelen işaretin seslimi yoksa sessizmi harflere ait olduğu belirlenir. Anahtarlama çıkışındaki işarete ait dizi gerekiyorsa, bir kazanç parametresi ile kuvvetlendirildikten sonra ses yolu'nu ya da vocal tract'ı temsil eden bir sayısal filtreye girilir. Filtrenin çıkışından ise ses işareti elde edilir.

Şekil 4.3 esasen bir ses sentezi sisteminin anahatlarını da göstermektedir. Modern sayısal ses sentezleyici sistemleri bu yapıyı referans almışlardır.

5- LOGARİTMİK SES TANIMA

Ses tanıma olayını gerçekleştiren sistemlerde, sistemin kalite ve kapasitesine göre çeşitli algoritmalar kullanılmaktadır. Biz burada başta da belirttiğimiz gibi, bir lineer prediktör filtresi çıkışındaki hata katsayılarını kullanarak tanıma olayının bilgisayar simülasyonunu gerçekleştireceğiz. Tanıma işlemi anahtarlarıyla aşağıda, Şekil 5.1'de gösterilmiştir. Analiz aralığı N uzunluklu diziler olarak seçilmiş olup, bu aralıkların herbiri N örnekli bir pencereden geçirilmiştir. Bu çalışmada N=200, 256 ve 300 olarak üç ayrı analiz aralığı seçeneği kullanılmış, pencere fonksiyonu için de Hamming ve Hanning şeklinde iki ayrı pencere kullanılmıştır.



Şekil 5.1: (a) Kayıt modu (b) Tanıma modu

Bu pencere fonksiyonlarının matematiksel ifadeleri aşağıdaki gibidir.

Hamming window:

N tek için $w(n)=0.54\ 0.46\ \mathrm{Cos}\{2\pi n/(N-1)\}$, $-(N-1)/2 \leq n \leq (N-1)/2$

N çift için $w(n)=0.54\ 0.46\ \mathrm{Cos}\{2\pi(2n\ 1)/2(N-1)\}$, $-(N/2) \leq n \leq (N/2-1)$

Hanning window:

N tek için $w(n)=0.5\ 0.5\ \mathrm{Cos}\{2\pi n/(n\ 1)\}$, $-(N-1)/2 \leq n \leq (N-1)/2$

N Çift için $w(n)=0.5\ 0.5\ \mathrm{Cos}\{2\pi(2n\ 1)/2(n\ 1)\}$, $-(N/2) \leq n \leq (N/2-1)$

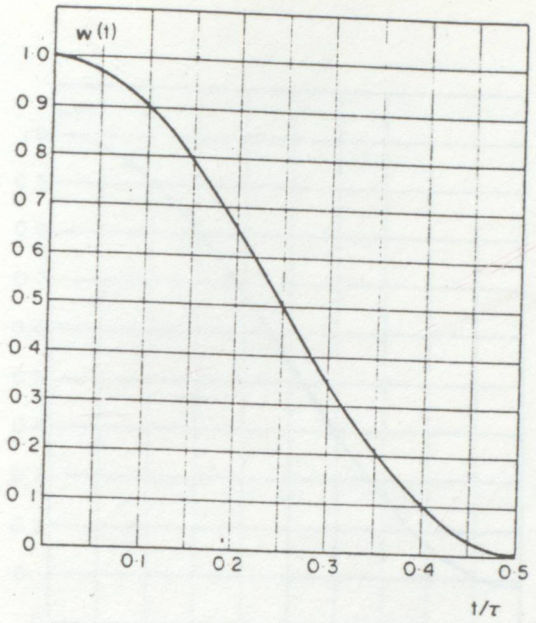
Bu fonksiyonlara ilişkin zaman ve frekans domenindeki genlik cevapları Şekil 2, 3, 4 ve 5'de verilmiştir. Bu arada şunu belirtmek gerekir, Şekil 2'dan Şekil 5'e kadar olan çizimler pencere fonksiyonlarının sürekli halleri içindir. Buradan ayrık hale geçmek mümkündür.

Girişine u(n) dizisi gelen pencerenin çıkışından u'(n) gibi yeni bir dizi elde edilir. Bu u'(n) dizisi bir LPC filtresi girişine verilir. LPC filtresinin çıkışından prediktör katsayıları ile birlikte, her analiz aralığına ait E(M) hata paternleri ve özilişki katsayıları elde edilir. E(M) hata paterni o aralığa ait toplam karasel hata olup, bölüm 3'de anlatılan $P_f(M)$ lineer prediktör hatasından başka bir şey değildir. O halde E(M)'in matematiksel ifadesini aşağıdaki gibi yazmak mümkündür.

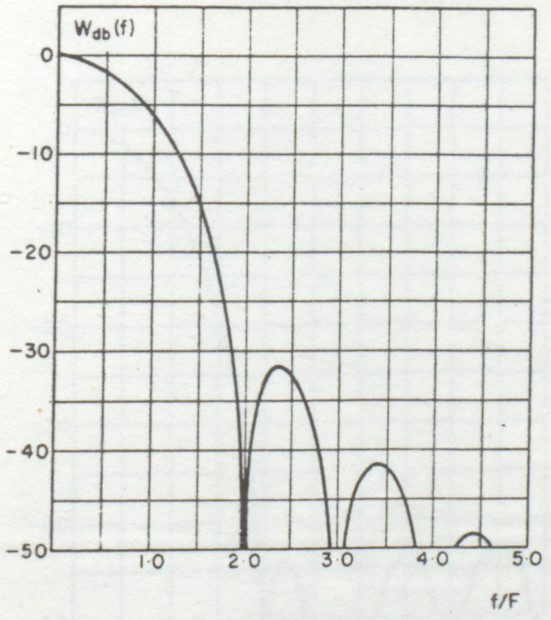
$$E(M) = \sum_{n=o}^{N-1} e^2(n)$$

$$= \sum_{n=o}^{N-1} \{u(n)-\tilde{u}(n)\}^2$$

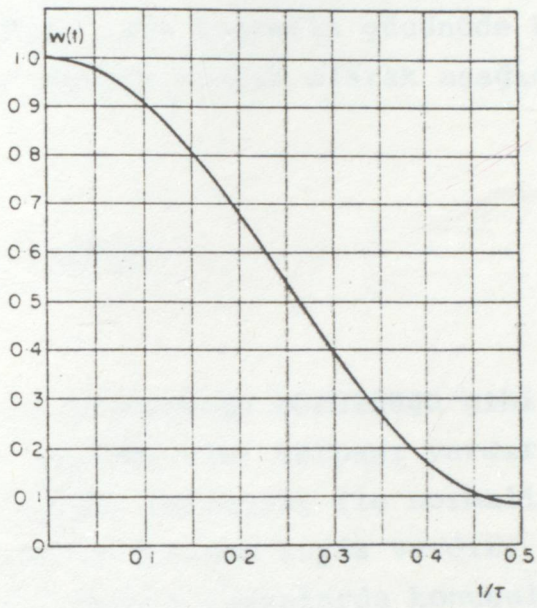
$$= r(o) \prod_{i=1}^{M} \{1-k^2(i)\}$$



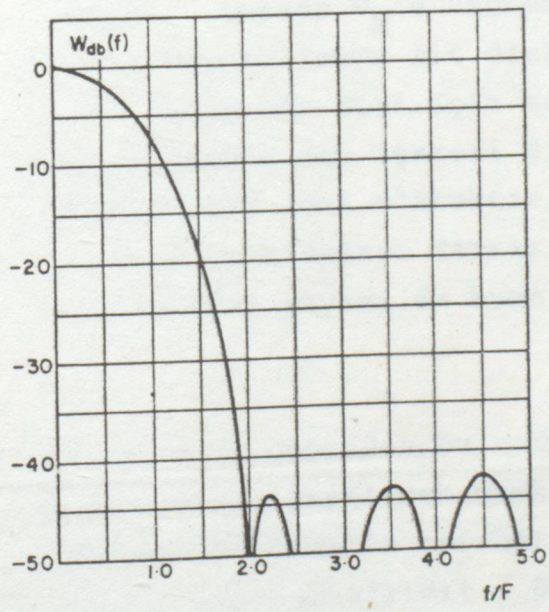
Şekil 5.2: Hanning pencere fonksiyonu



Şekil 5.3: Hanning pencere fonksiyonunun frekans-genlik cevabı



Şekil 5.4: Hamming pencere fonksiyonu



Şekil 5.5: Hamming pencere fonksiyonunun frekans-genlik cevabı.

Burada r(o) özilişki katsayısı olup aynı zamanda her analiz aralığı için işaretin gücünüde belirler. Özilişki katsayısını deterministik olarak aşağıdaki gibi ifade etmek mümkündür.

$$r(o) = \sum_{n=o}^{N-1-i} s^2(n) \qquad (5.3)$$

(5.2) ifadesinde görüldüğü gibi E(m) hata paterni ifadesinin başında r(o) çarpanı vardır. E(m) ifadesinin bu r(o) özilişki katsayısı ile normalize edilmesinde ses tanıma işlemi açısından fayda vardır. Çünkü bu şekilde aynı kelimenin farklı zamanlarda konuşulmasından kaynaklanan şiddet farkları ile bir derece elimine edilir. Bu normalizasyon işlemi yapılırsa, normalize hata paterni;

$$V_M \triangleq \frac{E(M)}{r(o)} \qquad (5.4)$$

ifadesiyle verilir. Burada V_M'e "Normalize Hata Paterni" denir. Konuşulan bir kelimeye ait dizinin uzunluğunu L ile gösterirsek, böyle bir dizi için analiz aralığı L/N olur. Dolayısıyla herbir ses işareti dizisi için elde edilen hata paterni de L/N adet olacaktır. Konuşulan farklı kelimeler için dizi uzunlukları farklı olacağından herbir diziye ait hata paterni sayısı da buna bağlı olarak değişecektir.

Tanıma işlemini doğrudan bu normalize hata paternlerini kullanarak gerçekleştirmek mümkündür. Bu, sistemin kalitesi açısından iyi olmakla beraber tanıma süresini uzatması açısından dezavantajlıdır. Bu yüzden bir adım daha giderek "Tanıma Paterni" dediğimiz ve H ile gösterdiğimiz yeni bir hata paterni daha tanımlayacağız. H tanıma

paterni ; normalize hata paternlerinin her analiz aralığı için iki katlarının dördüncü kuvveti alınıp her kelime dizisine ait analiz aralığı sayısı L/N boyunca toplamının alınmasıyla elde edilir. Yani matematiksel olarak aşağıdaki gibi ifade edilir :

$$H \overset{\Delta}{=} \sum_{i=1}^{L/N} \{2V_M(i)\}^4 \qquad (5.5)$$

Burada L/N analiz aralığı sayısı ve V_M normalize hata paternidir.

Bu ifadede, neden 2 çarpanının bulunduğu ve neden dördüncü kuvvetin alındığını şöylece izah edebiliriz : Her bir analiz aralığına ait normalize hata paterni için $|V_M| < 1$ olduğu gösterilebilir. Elde bulunan ve tanınmak için üzerinde işlemler yapılan ses dizileri için yine $|V_M| > 0.5$ olduğu gözlenmiştir. Bu durumda 2 ile çarptığımız takdirde 1'den büyük bir rakam elde edilecektir. Elde edilen bu 1'den büyük rakamın karesi yada dördüncü kuvveti gibi bir kuvveti alınırsa tanıma hassasiyeti açısından faydalı olacağı düşünülmüştür. Burada biz dördüncü kuvvetini tercih ettik.

Tanınması istenen her bir ses yada kelime paterni için bu şekilde bir H değeri elde edilir. Buraya kadar yapılan işler Şekil 1-a da sembolize edilmiştir. Dikkat edilirse bu analiz işlemleri "Kayıt" ve "Tanıma" modları diyebileceğimiz her iki durum için de söz konusudur.

Kayıt modu dediğimiz mod, örnek kelimelerin analiz edilerek sisteme tanıtılmasıdır. Bir insanın bir diğer insanın konuşmasını nasıl tanıdığını düşünürsek buradaki kayıt işleminin neden gerektiği ortaya çıkar. İnsan daha önce duyduğu bir sesi, sahibini görmemekle beraber bir telefon

konuşması esnasında tanıyabilir. Ancak daha önce kimden geldiğini öğrenmediği ve duymadığı bir sesi tanıyamaz. Buda bize gösterirki tanıma için önce bir öğrenme devresi gerekir. İşte bu öğrenme devresini biz burada "Kayıt" modu diye adlandırdık. Bizim çalışmamızda sisteme öğretilip kaydedilecek olan parametre ise, herbir kelimeye ait ses dizisi için bir adet olan tanıma paterni dediğimiz H' değerleridir. (5.5) ifadesiyle de bunun nasıl elde edileceği belirlidir.

Tanıma modu ise, adından anlaşılacağı üzere sisteme herhangibir anda girilen sesin tanınması için yapılan çalışmadır. Yukarda da belirttiğimiz gibi hem kayıt modu için, hem tanıma modu için Şekil 5.1.a'da blok diyagram halinde verilen analiz işlemleri yapılarak H tanıma paternleri elde edilir.

Şimdi tanıma işleminin nasıl gerçekleştiğini inceleyelim. Tanıma esasen bir karşılaştırma işlemidir. Yani sisteme daha önceden öğretilen ve kaydedilen birtakım parametrelerin, sisteme tanınmak üzere yeni girilen sese ait parametrelerle kıyas edilerek bir sonuca varılmasıdır. Bu karşılaştırma süresi aynı zamanda tanıma süresinin de bir miktarını işgal edeceğinden karşılaştırılacak parametre sayısı arttıkça tanıma süresi de artar. Toplam tanıma süresi, (parametre analiz süresi) artı (parametre karşılaştırma süresi)olduğundan çeşitli durumlar düşünülerek optimum bir çözüm elde edilebilir. Bizim çalışmamızda karşılaştırılacak olan parametre herbir kelime için 1 adet H tanıma paternidir. Çalışmamızda tanınmaya çalışılacak olan kelimeler "1" den "10" kadar rakamların Türkçe söylenmelerinden elde edilen kelimelerdir. Yani "Bir, iki,..., on" dur. Karşılaştırılacak parametre sayısı kelime başına yalnız bir adet ve kütüphanedeki kayıtlı kelimeler on adet olduğundan analiz işlemlerinden sonraki karşılaştırma süresi toplam tanıma

süresinin çok az bir miktarını işgal edecektir.

Karşılaştırma işleminin ne olduğunu ise şöyle izah edebiliriz: Sisteme daha önceden kaydedilen tanıma paternlerini H_i ile, tanıma işlemi esnasında elde edilen parametreyi ise $\hat{H}_j$ ile gösterelim. Kütüphanede on adet kelime bulunduğuna göre on adet kayıtlı H_i değeri olduğunu söylemek mümkündür. Bundan sonraki açıklamalarımızda H_i'lerden herbiri için referans tanıma paterni diye bahsedeceğiz. H_j'leri ise bilinmeyen tanıma paterni diye belirteceğiz.

Referans ve bilinmeyen tanıma paternlerini kullanarak bir uzaklık fonksiyonu tanımlamak mümkündür. Bu uzaklık fonksiyonu herbir referans paternin, bilinmeyen paterne oranının logaritması olarak tarif edilmiştir. Yani;

$$D(i,j) \triangleq abs\{\log(H_i/\hat{H}_j)\} \quad , i=1,2,\ldots,10 \qquad (5.6)$$

Burada D(i,j) uzaklık fonksiyonu, abs ise mutlak değer fonksiyonunu gösterir.

Neden böyle bir uzaklık fonksiyonun tanımlandığı kolayca görülebilir. İdeal halde, yani dış etkenlerin gözardı edilmesi halinde, eğer tanınmak üzere sisteme girilen kelime kütüphanedeki i. kelimeye karşılık geliyorsa $\hat{H}_j=H_i$ olacaktır. Uzaklik fonksiyonunun aldığı değer ise sıfır olacaktır. Yani;

$$D(i,j)=0 \quad , H_i=\hat{H}_j \text{ için}$$

olur. O halde bir genelleme olarak patikte şunu söylemek mümkündür. D(i,j) fonksiyonu, bilinmeyen paternin referans paternine eşit olması halinde minimum değerini alır. Yada başka bir deyişle, sisteme tanınmak üzere gelen ve $\hat{H}_j$ bilinmeyen paternine sahip olan kelime D(i,j) uzaklığını

minimum yapan ve kütüphanedeki H_i referans paternine sahip olan i. kelimedir.

Son cümle ile D(i,j) uzaklık fonksiyonunu kullanarak tanıma olayını nasıl gerçekleştireceğimizi özetlemiş olduk. Görüldüğü gibi tanınmak üzere sisteme girilen ve H_j bilinmeyen paternine sahip kelime, kütüphanedeki referans paternlerinden hangisiyle minimum bir uzaklığa sahipse o kelimeye eştir şeklinde bir sonuç kararı verilir.

Pratikte hiçbir zaman aynı kelimeye ait referans paterniyle, bilinmeyen paternlerinin eşit olması mümkün değildir. En ideal halde bile aralarındaki fark ancak belli bir minumuma inebilir. Bunun birçok nedeni vardır. Bunları ; kelimeyi söyleyen kişinin değişmesi, ortam gürültüsü farklı zamanlarda söylemekten kaynaklanan şiddet farklılıkları vs. gibi sıralayabiliriz. Bu bozucu etkenlere karşı etkisini azaltmak üzere önlem almak mümkündür. Ancak bunlarda herbiri için ayrı bir tasarım gerektirir. Biz bu etkilerden sadece şiddet farklılıklarının etkisini azalttık. Bunu ise normalizasyonla yaptığımızı belirtmiştik.

6- SİMÜLASYON ve SONUÇLARI

6.1. Tanıtım

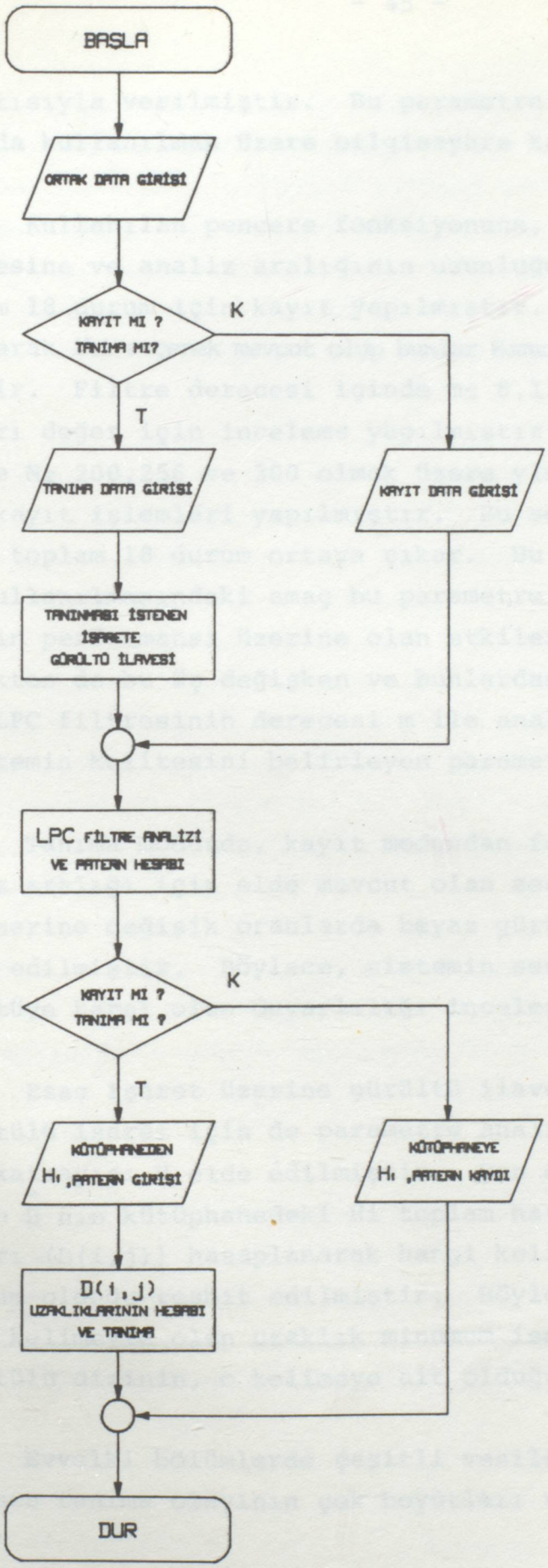
Simülasyon işlemi, geçen 5 bölümde anlalılan teori ve yöntemler esas alınarak bir VAX-11 sisteminde yüksek seviyeli bir programlama dili olan FORTRAN-77 kullanılarak gerçekleştirilmiştir.

İlk olarak bir erkek kişinin söylediği bir'den on'a kadar rakamların Türkçe söylenişleri, 10 kHz'lik bir örnekleme frekansıyla örneklenerek, elde edilen sayısal değerler sisteme kaydedilmiştir. Sonra da bu sayısal datalar işlenerek sonuca gidilmiştir.

Ses tanıma işleminin yapısı itibarıyla çalışma iki aşamada gerçekleştirilmiştir. Tahmin edileceği gibi ilk aşama kayıt modunu, ikinci aşama ise tanıma modunu gösterir Program, yazılım itibarıyla bir bütün halindedir. Ancak çalışılan moda uygun olarak kendi içinde dallanmalar mevcuttur. Programın akış diyagramı şekil 6.1. de verilmiştir. Programın kendisi ise toplu olarak EK-I de verilmiştir.

Elimizdeki ses örneği sadece bir kişiden ve bir defada alındığı için yöntemin çalışma performansı, eldeki mevcut dataya dışardan gürültü ilave edilerek tesbit edilmeye çalışılmıştır.

İlk aşamada yukarda da belirttiğimiz gibi elde mevcut olan data işlenerek on adet toplam hata parametresi hesaplanmıştır. Toplam hata parametresi bölüm 5'de (5.5)



Şekil 6.1: Simülasyon programının akış şeması

bağıntısıyla verilmiştir. Bu parametreler daha sonra tanıma modunda kullanılmak üzere bilgisayara kaydedilmiştir

Kullanılan pencere fonksiyonuna, LPC filtresinin derecesine ve analiz aralığının uzunluğuna bağlı olarak toplam 18 durum için kayıt yapılmıştır. Pencere fonksiyonu olarak iki seçenek mevcut olup bunlar Hamming ve Hanning Pencereleridir. Filtre derecesi içinde m= 8,12 ve 16 olmak üzere üç ayrı değer için inceleme yapılmıştır. Analiz aralığı içinse N= 200,256 ve 300 olmak üzere yine üç ayrı değer için kayıt işlemleri yapılmıştır. Bu seçenekler düşünüldüğünde toplam 18 durum ortaya çıkar. Bu değişik seçeneklerin kullanılmasındaki amaç bu parametrelerin ses tanıma sisteminin performansı üzerine olan etkilerini incelemektir. Gerçekten de bu üç değişken ve bunlardan özellikle son ikisi yani LPC filtresinin derecesi m ile analiz aralığı uzunluğu N sistemin kalitesini belirleyen parametrelerdir.

Tanıma modunda, kayıt modundan farklı olarak, her analiz aralığı için elde mevcut olan ses işaretine ait datalar üzerine değişik oranlarda beyaz gürültü (White Noise) ilave edilmiştir. Böylece, sistemin ses şiddeti açısından gürültüye karşı olan duyarlılığı incelenmiştir.

Esas işaret üzerine gürültü ilave edilmesiyle oluşan gürültülü işaret için de parametre analizi yapılarak toplam hata katsayısı $\hat{H}$ elde edilmiştir. Son olarak da, elde edilen bu $\hat{H}$ nın kütüphanedeki Hi toplam hatalarıyla olan uzaklıkları (D(i,j)) hasaplanarak hangi kelimeye ait uzaklığın minimum olduğu tesbit edilmiştir. Böylece kütüphanedeki hangi kelimeyle olan uzaklık minumum ise ; sisteme girilen gürültülü dizinin, o kelimeye ait olduğuna karar verilmiştir.

Evvelki bölümlerde çeşitli vesilelerle belirttiğimiz gibi ses tanıma olayının çok boyutları vardır. Bizim burda

yaptığımız çalışma ile elde edilen sonuçlar zaten tahmin edilen birtakım parametre değerlerinin ne olması lazım geldiğinin bir nevi objektif tasdiki şeklinde olmuştur. Çünkü herşeyden önce işaret üzerine eklenen gürültü beyaz gürültü olup pratik uygulama için fazla ideal bir seçimdir. Ancak dediğimiz gibi maksadımız sistemin parametrelere karşı olan hassasiyetini ölçmek olduğundan bizim için yeterli olmuştur.

Ses tanıma sistemlerinde, sesin farklı kişiler tarafından yada farklı zamanlarda söylenmesinden kaynaklanan iki problem olduğu daha önce vurgulanmıştır. Bunlardan birincisi sesdeki şiddet farklılığı, ikincisi ise süre farklılığı idi. Bu çalışmada şiddet farklılığından doğan sapmalar enerji normalizasyonuyla bir derece elimine edilmiştir. Ancak zaman bakımından meydana gelebilecek değişiklikler incelenmemiştir. Profesyonel sistemlerde bir takım dinamik programlama algoritmaları kullanılarak zamandaki değimlerin etkisini de minimuma indirgemek mümkündür. Burada dikkate alınmamıştır.

Tanıtım açısından son olarak şunları söyleyebiliriz ki ; yapılan çalışma ses tanıma sistemlerinde hangi parametrelerin nasıl etkidiğini görmek açısından faydalı olmuştur

6.2 Sonuçlar

Bu çalışmada ses şiddetinin gürültüyle, yada herhangi bir etkisiyle değişmesinin tanıma olayı üzerine etkilerini incelemeyi hedeflediğimiz için yukarda belirttiğimiz 18 ayrı durum için işaret-gürültü oranını değiştirerek inceleme yaptık. İncelenilen parametreler şunlardır :

N,(Analiz aralığı) : 200, 256, 300
M,(LPC Filtre derecesi) : 8, 12, 16
Pencere Fonksiyonu : Hamming, Hanning

Yapılan incelemeler sonucunda en uygun analiz aralığının N= 200, pencere fonksiyonunun Hanning ve filtre derecesinin ise M= 16 olduğu görülmüştür. Yani parametreler bu değerlere set edildiğinde tanıma işleminin başarısı en iyi olmaktadır. İnceleme yapılırken herbir durum için işaret-gürültü oranı (S/N), 10 ile 1 aralığında değiştirilmiştir.

Analiz aralığının N= 256 ve N= 300 olduğu durumlarda diğer parametreler ne olursa olsun sistem on kelimeden en az birini çok büyük işaret-gürültü oranlarında dahi (1000 gibi) tanıyamaz hale geldiği için bu bölümün geri kalan kısmında sadece N= 200 için incelemeye devam edeceğiz. Aşağıdaki bölümlerde N= 200 için diğer parametrelerin durumu incelenmiştir.

6.2.1. Pencere fonksiyonunun seçimi

Tanıma başarısının az bir farkla dahi olsa Hanning penceresinin seçilmesi halinde yüksek olduğunu yukarda söylemiştik. Şekil 6.2 de Hanning penceresi için, Şekil 6.3 de ise Hamming penceresi için alınan sonuçlar toplu halde verilmiştir. Şekil 6.4 ve Şekil 6.5 de ise sırasıyla Hanning ve Hamming pencereleri için bu sonuçlar grafiğe dökülmüştür. M'lerin aynı olduğu durumlar karşılıklı olarak kıyas edildiğinde herbir durum için (Yani M=8, M=12, M=16 olmak üzere üç ayrı durum için) Hanning penceresi kullanılması halinde tanıma başarısının daha yüksek olduğu görülür. Mesela M= 12 olması halinde Hanning penceresiyle S/N = 4 için sistem bütün kelimeleri tanıyabildiği halde (tanıma %100), Hamming penceresiyle on kelimeden bir tanesini yanlış tanımıştır. Yine grafikten görüleceği gibi Hamming penceresiyle aynı yüzde yüzlük bir tanıma başarısı için minimum işaret-gürültü oranı S/N=5 olmalıdır.

M=8
N=200

(S/N) \ KELIME	1	2	3	4	5	6	7	8	9	10
8	+	+	+	+	+	+	+	+	+	+
7	+	+	+	+	+	+	+	+	+	+
6	+	+	+	+	−	+	+	+	+	+
5	−	+	+	+	−	+	+	+	+	+
4	−	+	+	+	−	+	+	+	+	+
3	−	+	+	−	−	+	−	+	+	+
2	−	−	−	−	−	−	−	+	−	+
1	−	−	−	−	−	−	−	+	−	+

M=12
N=200

(S/N) \ KELIME	1	2	3	4	5	6	7	8	9	10
8	+	+	+	+	+	+	+	+	+	+
7	+	+	+	+	+	+	+	+	+	+
6	+	+	+	+	+	+	+	+	+	+
5	+	+	+	+	+	+	+	+	+	+
4	+	+	+	+	+	+	+	+	+	+
3	+	+	+	+	−	+	+	+	+	+
2	−	−	+	+	−	+	−	+	−	+
1	−	−	−	−	−	−	−	+	−	+

M=16
N=200

(S/N) \ KELIME	1	2	3	4	5	6	7	8	9	10
8	+	+	+	+	+	+	+	+	+	+
7	+	+	+	+	+	+	+	+	+	+
6	+	+	+	+	+	+	+	+	+	+
5	+	+	+	+	+	+	+	+	+	+
4	+	+	+	+	+	+	+	+	+	+
3	+	+	+	+	−	+	+	+	+	+
2	−	−	+	+	−	+	+	+	−	+
1	−	−	−	+	−	−	−	+	−	+

Şekil 6.2: Hanning penceresi için alınan sonuçlar

M=8
N=200

(S/N) \ KELIME	1	2	3	4	5	6	7	8	9	10
8	+	+	+	+	+	+	+	+	+	+
7	+	+	+	+	+	+	+	+	+	+
6	+	+	+	+	−	+	+	+	+	+
5	−	+	+	+	−	+	−	+	+	+
4	−	−	+	−	−	+	−	+	+	+
3	−	−	+	−	−	+	−	+	−	+
2	−	−	−	−	−	−	−	+	−	+
1	−	−	−	−	−	−	−	+	−	−

M=12
N=200

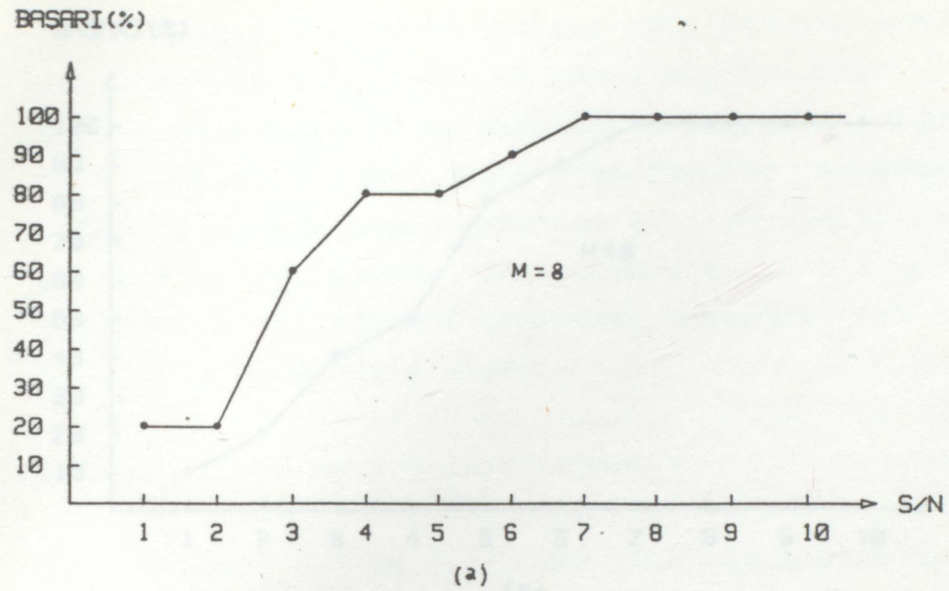
(S/N) \ KELIME	1	2	3	4	5	6	7	8	9	10
8	+	+	+	+	+	+	+	+	+	+
7	+	+	+	+	+	+	+	+	+	+
6	+	+	+	+	+	+	+	+	+	+
5	+	+	+	+	+	+	+	+	+	+
4	+	−	+	+	−	+	+	+	+	+
3	−	−	+	+	−	+	+	+	−	+
2	−	−	+	+	−	−	−	+	−	+
1	−	−	−	−	−	−	−	+	−	+

M=16
N=200

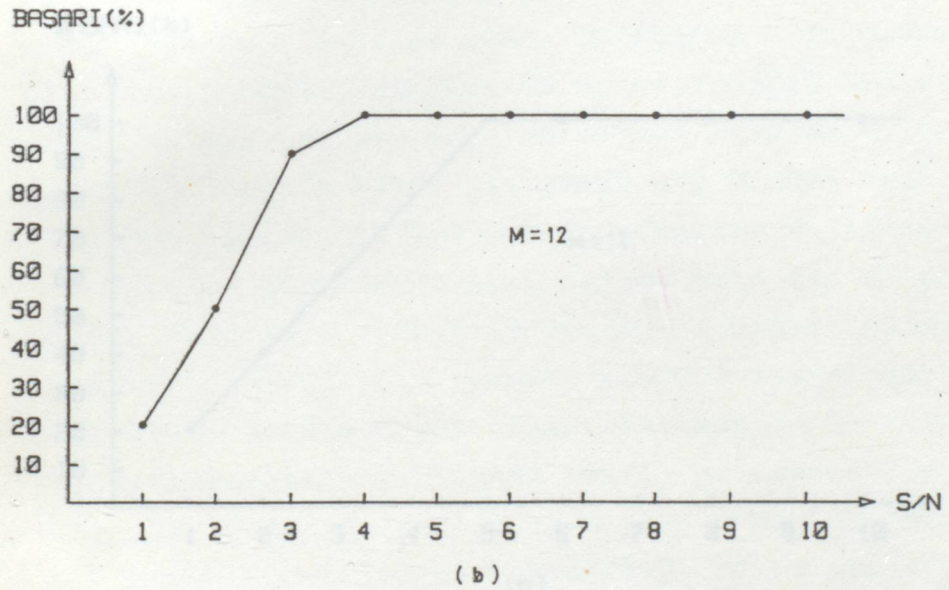
(S/N) \ KELIME	1	2	3	4	5	6	7	8	9	10
8	+	+	+	+	+	+	+	+	+	+
7	+	+	+	+	+	+	+	+	+	+
6	+	+	+	+	+	+	+	+	+	+
5	+	+	+	+	+	+	+	+	+	+
4	+	+	+	+	−	+	+	+	+	+
3	+	+	+	+	−	+	+	+	−	+
2	−	−	+	+	−	+	+	+	−	+
1	−	−	−	−	−	−	−	+	−	+

Şekil 6.3: Hamming penceresi için alınan sonuçlar

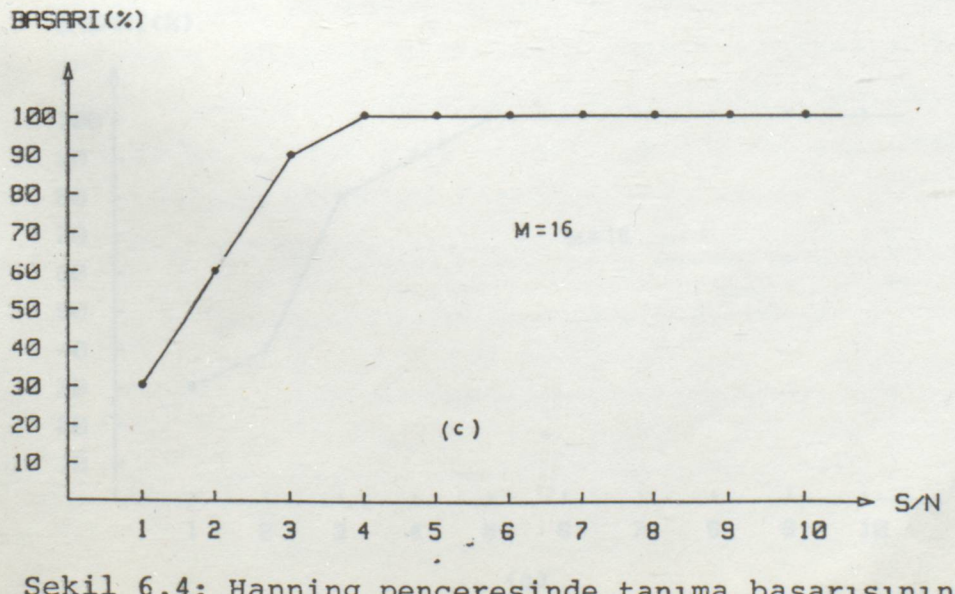
HANNİNG PENCERESİ İÇİN



(a)



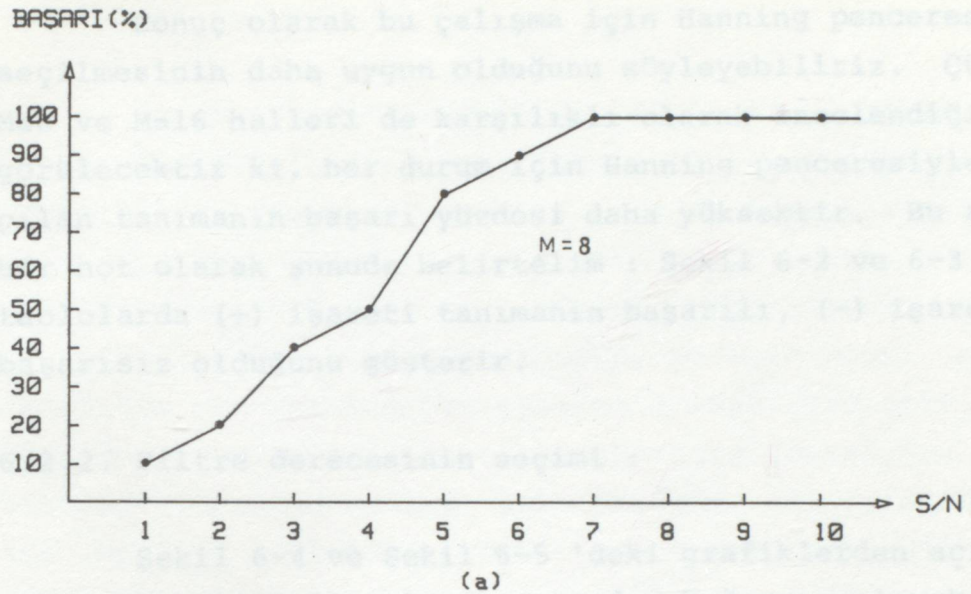
(b)



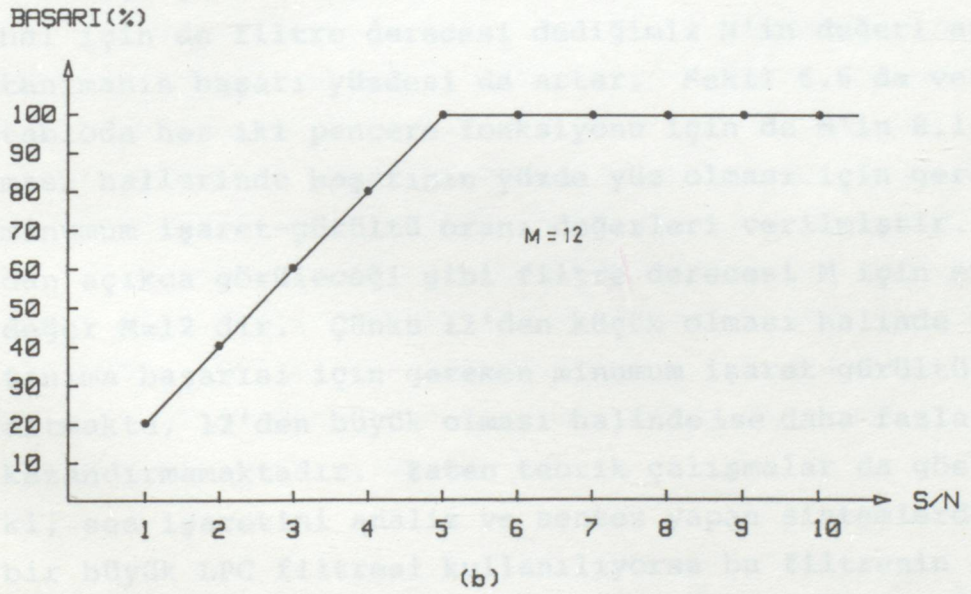
(c)

Şekil 6.4: Hanning penceresinde tanıma başarısının gürültü ile değişimi.

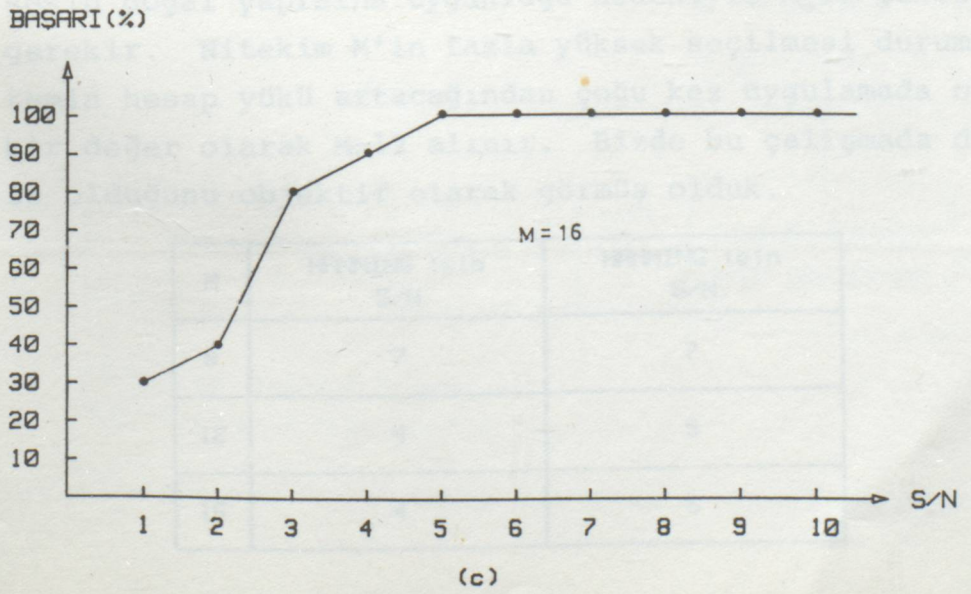
HAMMİNG PENCERESİ İÇİN



(a)



(b)



(c)

Şekil 6.5: Hamming penceresinde tanıma başarısının gürültü ile değişimi.

Sonuç olarak bu çalışma için Hanning penceresinin seçilmesinin daha uygun olduğunu söyleyebiliriz. Çünkü M=8 ve M=16 halleri de karşılıklı olarak incelendiğin de görülecektir ki, her durum için Hanning penceresiyle yapılan tanımanın başarı yüzdesi daha yüksektir. Bu arada bir not olarak şunuda belirtelim : Şekil 6-2 ve 6-3 deki tablolarda (+) işareti tanımanın başarılı, (-) işareti ise başarısız olduğunu gösterir.

6.2.2. Filtre derecesinin seçimi :

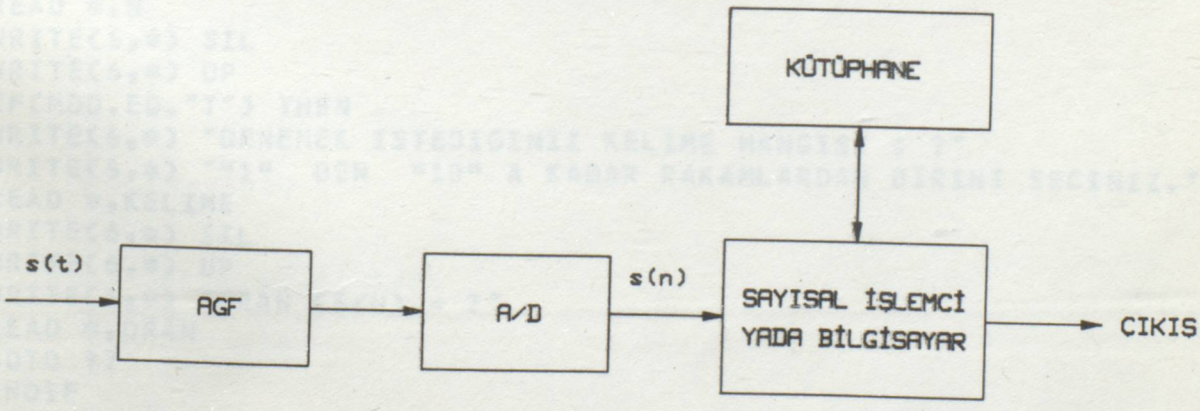
Şekil 6-4 ve Şekil 6-5 'deki grafiklerden açıkca görüleceği gibi kullanılan pencereden bağımsız olarak her iki hal için de filtre derecesi dediğimiz M'in değeri arttıkça tanımanın başarı yüzdesi de artar. Şekil 6.6 da verilen tabloda her iki pencere fonksiyonu için de M'in 8,12,16 olması hallerinde başarının yüzde yüz olması için gereken minumum işaret-gürültü oranı değerleri verilmiştir. Buradan açıkca görüleceği gibi filtre derecesi M için en uygun değer M=12 dir. Çünkü 12'den küçük olması halinde yüzde yüz tanıma başarısı için gereken minumum işaret-gürültü oranı artmakta, 12'den büyük olması halinde ise daha fazla birşey kazandırmamaktadır. Zaten teorik çalışmalar da göstermiştirki; ses işaretini analiz ve sentez yapan sistemlerde eğer bir büyük LPC filtresi kullanılıyorsa bu filtrenin derecesini sesin doğal yapısına uygunluğu nedeniyle M≤12 şeklinde seçmek gerekir. Nitekim M'in fazla yüksek seçilmesi durumunda sistemin hesap yükü artacağından çoğu kez uygulamada optimum bir değer olarak M=12 alınır. Bizde bu çalışmada durumun böyle olduğunu objektif olarak görmüş olduk.

M	HANNING için S/N	HAMMING için S/N
8	7	7
12	4	5
16	4	5

Şekil 6.6 : %100 başarı için gerekli minimum işaret-gürültü oranları

6.3. Genel sonuç

Bu çalışmadaki hedefimiz LPC filtresi kullanan ses tanıma sistemlerindeki parametrelerin genel davranış karakteristiğini ortaya çıkarmaktır. Bunda başarılı olduğumuzu zannediyorum. Çünkü teorik olarak bilinen bazı gerçeklerin pratik uygulamada bir nevi gözlemi yapılmıştır. Sonuç olarak pratikte de kullanılabilecek, parametreleri optimum değerlere set edilmiş bir çalışma ortaya çıkmıştır. EK-1 de verilen program, parametrelerin pencere fonksiyonu için Hanning'e, analiz aralığı için N=200'e, filtre derecesi için ise M=12'ye set edilirse bir ses tanıma sisteminin yazılım kısmı olarak kullanılabilir. Tahmin edileceği gibi bir ses tanıma sisteminin tamamını gerçekleştirmek için ilave bazı donanımlar gerekir. Çünkü ses işaretinin sayısal diziler haline gelebilmesi için önce bir filtre den geçmesi (tipik olarak 4 kHz band genişliğinde) daha sonra da bir anolog-dijital çeviriciden (A/D) geçirilmesi gerekir. Elde edilen bu dizinin bilgisayara kaydedilmesini sağlayacak gerekli arabirimler de sağlandığı takdirde ses tanımı sistemi bir bütün olarak elde edilmiş olur. Şekil 6-7'de ses tanıma sisteminin donanım olarak nelerden oluşacağı çok genel bir çizimle gösterilmiştir.



Şekil 6.7: Bir ses tanıma sisteminin genel donanımı.

EK-I

```
C*************************************************************************
C*    PROGRAMIN AMACI:LPC KULLANAN BIR SES TANIMA SISTEMININ          ****
C*                    SIMULASYONU                                     ****
C*    HAZIRLAYAN     :MUSTAFA TURFAN                                  ****
C*    PROGRAM DILI   :FORTRAN 77                                      ****
C*************************************************************************
      PROGRAM PRED
      DIMENSION ER(500),W(300),ERN(1000),H(10),HX(10),G(6000)
      DIMENSION D(10,10),RO(500),Y(6000),Z(6000),X(49000)
      DIMENSION P(20),Q(20),R(20),A(20,20),E(20)
      CHARACTER*20 NORMER,SIL,UP,MOD,PENCERE
      CHARACTER*7 VERSION
      INTEGER KELIME,T,ORAN,LENGTH
      REAL NOISE
      SIL=CHAR(27)//"[2J"
      UP=CHAR(27)//"[1;1H"
      PI=4.*ATAN(1.)
      WRITE(6,*) SIL
      WRITE(6,*) UP
      WRITE(6,*) "HANGI MODU ISTERSINIZ ?"
      WRITE(6,*) "KAYIT ICIN "K" YA BASINIZ."
      WRITE(6,*) "TANIMA ICIN "T" YE BASINIZ."
      READ(5,40) MOD
      WRITE(6,*) SIL
      WRITE(6,*) UP
      WRITE(6,*) "HANGI PENCERE"YI ISTERSINIZ ?"
      WRITE(6,*) "HAMMING ICIN "H" YA BASINIZ."
      WRITE(6,*) "COS ICIN "C" YE BASINIZ."
      READ(5,40) PENCERE
      WRITE(6,*) SIL
      WRITE(6,*) UP
      WRITE(6,*) "M = ?"
      READ *,M
      WRITE(6,*) SIL
      WRITE(6,*) UP
      WRITE(6,*) "N = ?"
      READ *,N
      WRITE(5,*) SIL
      WRITE(6,*) UP
      IF(MOD.EQ."T") THEN
      WRITE(6,*) "DENEMEK ISTEDIGINIZ KELIME HANGISI : ?"
      WRITE(6,*) ""1"  DEN  "10" A KADAR RAKAMLARDAN BIRINI SECINIZ."
      READ *,KELIME
      WRITE(6,*) SIL
      WRITE(6,*) UP
      WRITE(6,*) "ORAN (S/N) = ?"
      READ *,ORAN
      GOTO 77
      ENDIF
      WRITE(6,*) SIL
      WRITE(6,*) UP
      WRITE(6,*) "NORMALIZE HATA FILENAME (H) : ?"
      READ (5,40) NORMER
      WRITE(6,*) SIL
      WRITE(6,*) UP
      FORMAT(A30)
      WRITE(6,*) "LUTFEN BEKLEYINIZ !"
      IF(PENCERE.EQ."C") THEN
```

```
      DO 66 I=-N/2,N/2-1
      W(N/2+I+1)=0.5+0.5*COS(2*PI*REAL(N)/(REAL(N)+1))
66    CONTINUE
      GOTO 222
      ENDIF
      DO 50 I=-N/2,N/2-1
      W(N/2+I+1)=0.54+0.46*COS(PI*(2.*REAL(I)+1.)/(REAL(N)-1.))
50    CONTINUE
222   OPEN(UNIT=8,FILE='AT.DAT',STATUS='OLD')
      READ(8,*) (X(I),I=1,48000)
      CLOSE(UNIT=8)
      IF(MOD.EQ.'K') THEN
      L=48000
      LENGTH=0
      GOTO 55
      ENDIF
      IF(KELIME.EQ.1) THEN
      L=5400
      LENGTH=0
      GOTO 99
      ENDIF
      IF(KELIME.EQ.2) THEN
      L=4600
      LENGTH=5400
      GOTO 99
      ENDIF
      IF(KELIME.EQ.3) THEN
      L=4200
      LENGTH=10000
      GOTO 99
      ENDIF
      IF(KELIME.EQ.4) THEN
      L=5000
      LENGTH=14200
      GOTO 99
      ENDIF
      IF(KELIME.EQ.5) THEN
      L=4800
      LENGTH=19200
      GOTO 99
      ENDIF
      IF(KELIME.EQ.6) THEN
      L=4800
      LENGTH=24000
      GOTO 99
      ENDIF
      IF(KELIME.EQ.7) THEN
      L=4400
      LENGTH=28800
      GOTO 99
      ENDIF
      IF(KELIME.EQ.8) THEN
      L=5600
      LENGTH=33200
      GOTO 99
      ENDIF
      IF(KELIME.EQ.9) THEN
      L=5600
      LENGTH=38800
      GOTO 99
```

```
      ENDIF
      IF(KELIME.EQ.10) THEN
      L=3600
      LENTGH=44400
      ENDIF
99    IF(PENCERE.EQ.'C') GOTO 1
      IF(N.EQ.200) GOTO 2
      IF(N.EQ.256) GOTO 3
      IF(N.EQ.300) GOTO 4
2     IF(M.EQ.8) VERSION='H11.DAT'
      IF(M.EQ.12) VERSION='H13.DAT'
      IF(M.EQ.16) VERSION='H15.DAT'
      GOTO 455
3     IF(M.EQ.8) VERSION='H17.DAT'
      IF(M.EQ.12) VERSION='H19.DAT'
      IF(M.EQ.16) VERSION='H21.DAT'
      GOTO 455
4     IF(M.EQ.8) VERSION='H23.DAT'
      IF(M.EQ.12) VERSION='H25.DAT'
      IF(M.EQ.16) VERSION='H27.DAT'
      GOTO 455
1     IF(N.EQ.200) GOTO 5
      IF(N.EQ.256) GOTO 6
      IF(N.EQ.300) GOTO 7
5     IF(M.EQ.8) VERSION='H12.DAT'
      IF(M.EQ.12) VERSION='H14.DAT'
      IF(M.EQ.16) VERSION='H16.DAT'
      GOTO 455
6     IF(M.EQ.8) VERSION='H18.DAT'
      IF(M.EQ.12) VERSION='H20.DAT'
      IF(M.EQ.16) VERSION='H22.DAT'
      GOTO 455
7     IF(M.EQ.8) VERSION='H24.DAT'
      IF(M.EQ.12) VERSION='H26.DAT'
      IF(M.EQ.16) VERSION='H28.DAT'
455   J=315759
      GUC=0
      DO 444 I=1,N
      Y(I)=RAN(J)
      Z(I)=RAN(J)
      Y(I)=SQRT(-2*LOG(Y(I)))*COS(2*PI*Z(I))
444   CONTINUE
55    BOLUM=L/N
      KALAN=JMOD(L,N)
      IF(KALAN.NE.0) THEN
      BOLUM=(L-KALAN+N)/N
      ENDIF
      DO 125 D=0,BOLUM-1
      IF(MOD.EQ.'K') GOTO 555
      GUC=0
      DO 109 I=1,N
      GUC=GUC+X(I+LENGTH+D*N)*X(I+LENGTH+D*N)
109   CONTINUE
      NOISE=SQRT(GUC/N)/ORAN
      DO 301 I=1,N
      X(I+LENGTH+D*N)=X(I+LENGTH+D*N)+Y(I)*NOISE
301   CONTINUE
555   DO 119 J=1,N
      X(J+LENGTH+D*N)=X(J+LENGTH+D*N)*W(J)
119   CONTINUE
```

```
      R0(0+1)=0
      ER(0+1)=0
      DO 159 K=1,M
      DO 209 J=1,N-1-K
      R0(0+1)=R0(0+1)+X(J+LENGTH+N*0)*X(J+LENGTH+N*0)
209   CONTINUE
159   CONTINUE
      DO 309 I=1,M
      TOP1=0
      DO 409 J=1,N-1-I
      K=ABS(J-I)
      IF(K.EQ.0) TOP1=X(J+1+LENGTH+N*0)*X(1+LENGTH+N*0)+TOP1
      IF(K.NE.0) TOP1=X(J+1+LENGTH+N*0)*X(K+LENGTH+N*0)+TOP1
409   CONTINUE
      R(I)=TOP1
309   CONTINUE
      TOP2=0
      Q(1)=-(R(1)/R0(0+1))
      E(1)=(1-Q(1)**2)*R0(0+1)
      A(1,1)=Q(1)
      P(1)=Q(1)
      DO 509 I=2,M
      DO 609 J=1,I-1
      TOP2=TOP2+A(J,I-1)*R(I-J)
609   CONTINUE
      Q(I)=-(R(I-1)+TOP2)/E(I-1)
      E(I)=(1-Q(I)**2)*E(I-1)
      DO 709 K=1,I-1
      A(K,I)=A(K,I-1)+Q(I)*A(I-K,I-1)
      IF (K.EQ.I) A(K,I)=Q(I)
      IF (I.EQ.M) P(K)=A(K,M)
709   CONTINUE
509   CONTINUE
      P(M)=Q(M)
      ER(0+1)=E(M)
      ERN(0+1)=2*ER(0+1)/R0(0+1)
125   CONTINUE
      IF(MOD.EQ."K") GOTO 1500
      HX(KELIME)=0
      DO 1400 I=1,BOLUM
      HX(KELIME)=HX(KELIME)+ERN(I)**4
1400  CONTINUE
      OPEN(UNIT=9,FILE=VERSION,STATUS="OLD")
      READ(9,*) (H(I),I=1,10)
      CLOSE(UNIT=9)
      DO 300 I=1,10
      D(I,KELIME)=ABS(LOG(H(I)/HX(KELIME)))
300   CONTINUE
      T=1
      DO 400 I=2,10
      IF(D(I,KELIME).LT.D(T,KELIME)) T=I
400   CONTINUE
      WRITE(6,*) SIL
      WRITE(6,*) UP
      WRITE(6,*)                "TANINAN KELIME "
      PRINT *,                         T
      GOTO 2000
1500  DO 150 I=1,10
      H(I)=0
      IF(I.EQ.1) THEN
```

```
BOLUM2=5400/N
KALAN2=JMOD(5400,N)
GOTO 510
ENDIF
IF(I.EQ.2) THEN
BOLUM2=5400/N
BOLUM3=10000/N
KALAN2=JMOD(5400,N)
KALAN3=JMOD(10000,N)
GOTO 520
ENDIF
IF(I.EQ.3) THEN
BOLUM2=10000/N
BOLUM3=14200/N
KALAN2=JMOD(10000,N)
KALAN3=JMOD(14200,N)
GOTO 530
ENDIF
IF(I.EQ.4) THEN
BOLUM2=14200/N
BOLUM3=19200/N
KALAN2=JMOD(14200,N)
KALAN3=JMOD(19200,N)
GOTO 540
ENDIF
IF(I.EQ.5) THEN
BOLUM2=19200/N
BOLUM3=24000/N
KALAN2=JMOD(19200,N)
KALAN3=JMOD(24000,N)
GOTO 550
ENDIF
IF(I.EQ.6) THEN
BOLUM2=24000/N
BOLUM3=28800/N
KALAN2=JMOD(24000,N)
KALAN3=JMOD(28800,N)
GOTO 560
ENDIF
IF(I.EQ.7) THEN
BOLUM2=28800/N
BOLUM3=33200/N
KALAN2=JMOD(28800,N)
KALAN3=JMOD(33200,N)
GOTO 570
ENDIF
IF(I.EQ.8) THEN
BOLUM2=33200/N
BOLUM3=38800/N
KALAN2=JMOD(33200,N)
KALAN3=JMOD(38800,N)
GOTO 580
ENDIF
IF(I.EQ.9) THEN
BOLUM2=38800/N
BOLUM3=44400/N
KALAN2=JMOD(38800,N)
KALAN3=JMOD(44400,N)
GOTO 590
```

```
      IF(I.EQ.10) THEN
      BOLUM2=44400/N
      BOLUM3=48000/N
      KALAN2=JMOD(44400,N)
      KALAN3=JMOD(48000,N)
      GOTO 600
      ENDIF
510   IF(KALAN2.LT.N/2) BOLUM2=(5400-KALAN2)/N
      IF(KALAN2.GT.N/2) BOLUM2=(5400-KALAN2+N)/N
      DO 200 J=1,BOLUM2
      H(I)=H(I)+ERN(J)**4
200   CONTINUE
      GOTO 150
520   IF(KALAN2.LT.N/2) BOLUM2=(5400-KALAN2+N)/N
      IF(KALAN2.GT.N/2) BOLUM2=(5400-KALAN2)/N
      IF(KALAN3.LT.N/2) BOLUM3=(10000-KALAN3)/N
      IF(KALAN3.GT.N/2) BOLUM3=(10000-KALAN3+N)/N
      DO 210 J=BOLUM2,BOLUM3
      H(I)=H(I)+ERN(J)**4
210   CONTINUE
      GOTO 150
530   IF(KALAN2.LT.N/2) BOLUM2=(10000-KALAN2+N)/N
      IF(KALAN2.GT.N/2) BOLUM2=(10000-KALAN2)/N
      IF(KALAN3.LT.N/2) BOLUM3=(14200-KALAN3)/N
      IF(KALAN3.GT.N/2) BOLUM3=(14200-KALAN3+N)/N
      DO 220 J=BOLUM2,BOLUM3
      H(I)=H(I)+ERN(J)**4
220   CONTINUE
      GOTO 150
540   IF(KALAN2.LT.N/2) BOLUM2=(14200-KALAN2+N)/N
      IF(KALAN2.GT.N/2) BOLUM2=(14200-KALAN2)/N
      IF(KALAN3.LT.N/2) BOLUM3=(19200-KALAN3)/N
      IF(KALAN3.GT.N/2) BOLUM3=(19200-KALAN3+N)/N
      DO 230 J=BOLUM2,BOLUM3
      H(I)=H(I)+ERN(J)**4
230   CONTINUE
      GOTO 150
550   IF(KALAN2.LT.N/2) BOLUM2=(19200-KALAN2+N)/N
      IF(KALAN2.GT.N/2) BOLUM2=(19200-KALAN2)/N
      IF(KALAN3.LT.N/2) BOLUM3=(24000-KALAN3)/N
      IF(KALAN3.GT.N/2) BOLUM3=(24000-KALAN3+N)/N
      DO 240 J=BOLUM2,BOLUM3
      H(I)=H(I)+ERN(J)**4
240   CONTINUE
      GOTO 150
560   IF(KALAN2.LT.N/2) BOLUM2=(24000-KALAN2+N)/N
      IF(KALAN2.GT.N/2) BOLUM2=(24000-KALAN2)/N
      IF(KALAN3.LT.N/2) BOLUM3=(28800-KALAN3)/N
      IF(KALAN3.GT.N/2) BOLUM3=(28800-KALAN3+N)/N
      DO 250 J=BOLUM2,BOLUM3
      H(I)=H(I)+ERN(J)**4
250   CONTINUE
      GOTO 150
570   IF(KALAN2.LT.N/2) BOLUM2=(28800-KALAN2+N)/N
      IF(KALAN2.GT.N/2) BOLUM2=(28800-KALAN2)/N
      IF(KALAN3.LT.N/2) BOLUM3=(33200-KALAN3)/N
      IF(KALAN3.GT.N/2) BOLUM3=(33200-KALAN3+N)/N
      DO 260 J=BOLUM2,BOLUM3
      H(I)=H(I)+ERN(J)**4
```

```
      GOTO 150
580   IF(KALAN2.LT.N/2) BOLUM2=(33200-KALAN2+N)/N
      IF(KALAN2.GT.N/2) BOLUM2=(33200-KALAN2)/N
      IF(KALAN3.LT.N/2) BOLUM3=(38800-KALAN3)/N
      IF(KALAN3.GT.N/2) BOLUM3=(38800-KALAN3+N)/N
      DO 270 J=BOLUM2,BOLUM3
      H(I)=H(I)+ERN(J)**4
270   CONTINUE
      GOTO 150
590   IF(KALAN2.LT.N/2) BOLUM2=(38800-KALAN2+N)/N
      IF(KALAN2.GT.N/2) BOLUM2=(38800-KALAN2)/N
      IF(KALAN3.LT.N/2) BOLUM3=(44400-KALAN3)/N
      IF(KALAN3.GT.N/2) BOLUM3=(44400-KALAN3+N)/N
      DO 280 J=BOLUM2,BOLUM3
      H(I)=H(I)+ERN(J)**4
280   CONTINUE
      GOTO 150
600   IF(KALAN2.LT.N/2) BOLUM2=(44400-KALAN2+N)/N
      IF(KALAN2.GT.N/2) BOLUM2=(44400-KALAN2)/N
      IF(KALAN3.LT.N/2) BOLUM3=(48000-KALAN3)/N
      IF(KALAN3.GT.N/2) BOLUM3=(48000-KALAN3+N)/N
      DO 290 J=BOLUM2,BOLUM3
      H(I)=H(I)+ERN(J)**4
290   CONTINUE
150   CONTINUE
      OPEN(UNIT=9,FILE=NORMER,STATUS="NEW")
      WRITE(9,*) (H(I),I=1,10)
      CLOSE(UNIT=9)
2000  STOP
      END
```

KAYNAKLAR

1- OPPENHEIM and SCHAFER, "Digital Signal Processing," Printice Hall, 1978.

2- OPPENHEIM, "Applications of Digital Signal Processing," Printice Hall, Vol. 3, 1978.

3- JOHN MAKHOUL, "Lineer Prediction: A Tutorial Review," Proc. IEEE, Vol. 63, pp. 561-580, Aprb 1975.

4- JOHN D. MARKEL, " A Lineer Prediction Vocoder Simulation Based upon the Autocorelation Method," IEEE Trans. Acoust., Speech and Signal Process., Vol. ASSP-22, pp. 124-134, Apr. 1974.

5- F. ITAKURA, "Minimum Prediction Residüal Principle Applied to Speech Recognation," IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-23, pp. 67-72, Feb. 1975.

6- SIMON HAYKIN, " Introduction to Adaptive Filters," Macmillan Publishing Company, 1984.

ÖZGEÇMİŞ

1963-Giresun doğumlu olan yazar ilköğrenimini Giresun'da tamamladı. Orta ve Lise öğreniminine ise Samsun'da devam etti. 1985 yılında Bursa'da Uludağ Üniversitesinde Elektronik Mühendisliğini bitiren yazar yüksek lisans eğitimine de İstanbul'da Yıldız Üniversitesinde devam etti. Halen telekominikasyon cihazları üreten bir kuruluş olan Teletaş'ta araştırma-geliştirme mühendisi olarak çalışmaktadır.