T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

SAYISAL FİLTRE TASARIM YÖNTEMLERİ VE PERFORMANS ANALİZLERİ

YÜKSEK LİSANS TEZİ

Zeynep BATIK

Enstitü Anabilim Dalı : ELEKTRONİK VE BİLGİSAYAR EĞİTİMİ

Tez Danışmanı : Doç. Dr. Ali Fuat BOZ

Haziran 2011

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

SAYISAL FİLTRE TASARIM YÖNTEMLERI VE PERFORMANS ANALİZLERİ

YÜKSEK LİSANS TEZİ

Zeynep BATIK

Enstitü Anabilim Dalı : ELEKTRONİK VE BİLGİSAYAR EĞİTİMİ

Bu tez 29/06/2011 tarihinde aşağıdaki jüri tarafından Oybirliği ile kabul edilmiştir.

Prof. Dr. Abdullah Feriloğlu	Doç. Dr. Ali Fuat Boz	Yrd. Doç. Dr. İhsan Pehlivan
Jüri Başkanı	**Üye**	**Üye**

TEŞEKKÜR

Tez çalışmalarım esnasında karşılaştığım problemlerde yardımlarını esirgemeyen Sayın Yrd. Doç. Dr. Fahri VATANSEVER, Doç. Dr. Ali Fuat BOZ ve Arş. Gör. Sezgin KAÇAR'a teşekkürü bir borç bilirim. Ayrıca destekleriyle hep yanımda olan ve hiçbir fedakârlıktan kaçınmayan aileme sonsuz teşekkür eder, bu tezi onlara ithaf ederim.

Haziran 2011

İÇİNDEKİLER

SİMGELER VE KISALTMALAR LİSTESİ

Hz	: Hertz
x(n)	: Giriş işareti
H	: Dönüştürme operatörü
y(n)	: Çıkış işareti
ω_c	: Kesim frekansı
f_s	: Örnekleme frekansı
f(x)	: Uygunluk fonksiyonu
ω	: Frekans aralığı
db	: Desibel
GA	: Genetik Algoritma
FIR	: Sonlu Dürtü Yanıtı
IIR	: Sonsuz Dürtü Yanıtı
NP-Zor	: Non-deterministic Polynomial Time
Eİ	: En iyileme algoritmaları
AGF	: Alçak geçiren filtre
YGF	: Yüksek geçiren filtre
BGF	: Band geçiren filtre
BDF	: Band durduran filtre

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar Kelimeler: Genetik Algoritma, FIR filtreler, sayısal filtreler, MATLAB GUI, ASP.NET

Bu tezde Genetik Algoritma(GA) kullanılarak sayısal filtre tasarımı gerçekleştirilmiştir.

Genetik algoritma parametrelerinden popülasyon boyutu, uygunluk fonksiyonun ölçeklenmesi, seçim fonksiyonu, mutasyon fonksiyonu, çaprazlama fonksiyonu ve çaprazlama oranı değişkenleri kullanılarak etkileri açıklanmıştır.

Sayısal filtre çeşitleri hakkında temel bilgiler verilerek ve FIR(sonlu dürtü yanıtı) filtrelerin yapıları ve filtre çeşitleri açıklanmıştır.

Genetik algoritma kullanılarak elde edilen FIR filtre katsayılarının hesaplanmasında ve elde edilen katsayı değerler ile filtre cevaplarının grafiksel olarak elde edilmesine yönelik iki farklı arayüz tasarlanmıştır. Bu arayüzlerden biri MATLAB GUI ile diğeri ise ASP.NET ile hazırlanmıştır. İki farklı arayüzden elde edilen sonuçlar karşılaştırılmıştır.

DİGİTAL FİLTER DESIGN METHODS AND PERFORMANCE ANALYSISES

SUMMARY

Keywords**:** Genetic algorithm, finite impuls response filter, digital filters, MATLAB GUI, ASP.NET

In this thesis; designing of a numerical fitler by using Genetic Algorithm(GA) was performed.

Population size, scaling of the fitness function, selection function, mutation function, cross-over function, and cross-over rate varieties which are all of the genetic algorithm parameters were used and their effects were explained.

Some information about numerical filters were given and structure of FIR(finite impulse response) and types of filters were explained.

Two different interface were designed for calculation of the coefficient of the FIR filters which were obtained by using Genetic algorithm and for obtaining the filter response graphically by using the obtained coefficient values. One of these interfaces was prepared by using MATLAB GUI and the other one by using ASP.NET. Results obtained from the both interfaces were compared.

BÖLÜM 1. GİRİŞ

Filtreleme, işaret ve görüntü işleme alanlarındaki temel süreçlerdendir. Filtreleme de karşılaşılan sorun istenilen bir sinyal elde edilirken, istenilmeyen sinyallerle ve parazitlerle birleşik olduğu için sinyalin istenmeyen sinyallerden yeterli düzeyde en iyi şekilde seçilmesi yetersiz kalmıştır. Bu yüzden elektronik filtreler, giriş terminalinden uygulanan sinyal çıkış terminalinden geçerken sadece istenilen sinyalin geçmesini sağlar. Geçişi sağlarken de sinyal frekansının azaltılması veya yükseltilmesi yoluyla gerçekleşir. Filtreler analog ve sayısal(dijital) olmak üzere ikiye ayrılır. Analog filtreler elektronikte kullanılan sinyal işlemenin temel yapı bloklarıdır ve sürekli değişen sinyallerle çalışırlar. Pasif doğrusal elektronik analog filtreler doğrusal türevsel denklemlerle gerçekleştirilir. Devre tasarlarken kapasitör, endüktör ve direnç gibi pasif devre elemanlarının birleşiminden oluşur. Analog filtreler genellikle telekominasyon ve dalga filtreleme de kullanılır. Sayısal filtreler sinyalin görünümünü artırmak veya azaltmak için örneklemiş, ayrık zamanlı sinyaller üzerinde matematiksel işlemleri, elektronik, bilgisayar bilimi ve matematik dalında gerçekleştirilen bir sinyaldir.

Sayısal filtreler genellikle digital sinyal işlemcisi tarafından entegre halinde tasarlandıkları için bir değişim olması gerektiği zaman sadece yazılım kısmında değişiklik olması gereklidir oysa analog filtreler de devre tasarımındaki elemanların değişmesi gerekir bu da maliyet ve zaman bakımından filtreler arası farkı gösterir. Analog filtrelerin çalışma noktası ve kararlığı sıcaklığa bağlı olduğu halde sayısal filtreleri sıcaklık etkilemez. Sayısal filtreleri gerçekleştirirken analog filtreden daha fazla gerçekleştirilme algoritmasına sahiptir. Sayısal filtreler analog filtrelere göre daha düşük frekanslı sinyallerle çalışabilirler. Analog filtrelerin sayısal filtre göre üstünlükleri ise enerjiyi geçirme, frekans aralık sınırı verilmemesi gibi sayılabilir.

Sayısal filtreler(süzgeçler) dürtü yanıtlarına göre sonsuz dürtü yanıtlı filtre (Infinite Impulse Response (IIR)) ve sonlu dürtü yanıtlı (Finite Impulse Response (FIR)) filtre şeklinde ikiye ayrılmaktadırlar.

Sonlu dürtü yanıtlı(FIR) filtrelerin sonsuz dürtü yanıtı (IIR) filtrelere göre avantajı doğrusal fazlı frekans yanıtlarıdır. FIR filtreler sabit grup gecikmesi sağlarlar. Sonsuz dürtü yanıtı (IIR) filtreler sonlu dürtü yanıtlı(FIR) filtrelere göre daha verimlidir çünkü daha az geciktirici elemana, toplayıcıya ve çoğaltıcıya ihtiyaç duyarlar. Keskin kesim frekanslı IIR filtrelerin FIR filtrelere göre tasarımındaki katsayıların hesaplanmasında doğruluk problemi önemlidir.

Tasarım aşamasında çıkışın ideale yakın olması istendiği için filtre derecesi artmakta ve filtrenin tasarımındaki çarpma ve toplama eleman sayısında artmalara neden olmaktadır. İstenen özelliklerde filtre tasarlayabilmek için pek çok yöntem bulunmaktadır. Filtre tasarımında (gerek IIR gerekse de FIR filtrede) temel amaç, filtre transfer fonksiyonunu oluşturan pay ve payda katsayılarının hesaplanması işlemine dayanmaktadır.

Sezgisel hesaplama yöntemlerinden biri olan ve sonuca farklı arama noktalarından değer üreterek ulaşan GA, klasik hesaplama yöntemlerinin yetersiz kaldığı veya farklı çözüm yollarının üretilmesinin istendiği uygulamalarda fazlaca kullanılmaktadır.

Literatürde sonlu dürtü yanıtı (FIR) katsayılarının hesaplanmasına yönelik birçok çalışmalar yer almaktadır.

2010 yılında Kedir-Talha ve arkadaşlarının yapmış olduğu çalışmada cenin kalp sinyalinin algılanması ve analizi elektronik ortamda izlenmesi hedeflenmiştir. Sonlu Dürtü Cevabı (FIR) uyarlanabilir filtreleri Genetik Algoritmayla gerçekleştirmişleridir. Bu gerçekleşmede ideal filtrenin oluşabilmesi için ikinci derece hata ve kesin yakınsamayı sağlayacak şekilde filtre katsayılarının minimizasyonu sağlanmıştır. GA'nın yapısında 8 bit ve 10 döngüyle birlikte çalışmıştır. Elde edilen sonuçlar Wiener, RLMS ve NLMS algoritmalarıyla elde

edilen sonuçlarla karşılaştırılmıştır. GA ile elde edilen grafik sonuçları ile ölçülen sinyal arasındaki farkın diğer algoritmalara göre daha az olduğu gösterilmiştir [1].

2010 yılında BAYILMIŞ tarafından yapılan dijital modülasyon tekniklerini MATLAB Builder NE ve MATLAB Web Figure araçlarının kullanıldığı bir arayüz tasarlanmıştır. Bu tasarımda dijital bilginin modülasyon teknikleri ile modülasyonu görsel bir şekilde anlatılmaktadır [2].

2010 yılında Dey ve arkadaşları tarafından yapılan çalışmada genetik algoritma aracılığıyla otomatik, hızlı ve en az hesaplama karmaşıklığına sahip bir FIR(sonlu dürtü cevabı) filtre tasarımı dinamik bir method geliştirilmiştir. Kodlanmış şemalarla verimli ve özel filtre katsayılarıyla çalışılmıştır. Algoritma filtre katsayılarından genom popülasyonu oluşturmuştur. Genetik algoritmayla elde edilen transfer fonksiyonla da en küçük kareleme yöntemi ve az-çok yaklaşımı gerçekleştirilmiştir. Bu filtrenin derecesi 33, 65 ve 129 olarak tasarlanmıştır. Tasarımda adım genişliğine dikkat edilmelidir. Her iki yaklaşımla da tasarlanan filtrenin frekans cevapları ve hata fonksiyonları gerçekleştirilmiştir. Karşılaştırma sonucun da en küçük kareleme yönteminden ideale yakın sonuçlar elde edilmiştir, bu noktada adım büyüklüğüne dikkat edilmelidir ve grafiksel olarak da cevaplar gösterilmiştir [3].

2010 yılında KAÇAR ve ÇANKAYA tarafından yapılan çalışmada Volterra serileri doğrusal olmayan sistemlerin analizinde yaygın bir şekilde kullanılır. 2007 yılında geliştirilen basitleştirilmiş algoritma kullanılarak, MATLAB programı ile kodlanmış ve MATLAB Builder NE ile .NET tabanlı bir arayüz tasarlanmıştır. Tasarımda Duffing denklemleriyle çalışılmıştır. Bu denklem ikinci dereceden doğrusal olmayan bir işlemdir. Denklemin birinci ve ikinci derece frekans cevabı elde edilmiştir [4].

2010 yılında Goyal ve arkadaşları tarafından yapılan çalışmada genetik algoritma kullanılarak düşük güçlü sonlu dürtü cevabı elde edilmiştir. Çalışmayı gerektiren sebep ise kablosuz iletişim sistemlerinin ve taşınabilir cihazların hızlı bir şekilde büyümesiyle beraberinde güç azaltma sorunu büyük bir problem olarak ortaya çıkmaktadır. İletişim sistemlerinin bazılarında gönderilen bilginin çözümü için sayısal sinyal işleme kullanılmıştır. Sayısal sinyal işleme yapı bloklarından sonlu

dürtü filtresi (FIR) seçilmiştir. FIR filtreler programlanabilir sayısal sinyal işlemcileri tarafından seri çoğaltıcı ve kaydedici birimler tarafından gerçekleştirilmiştir. İşlemci çoğaltıcı ve kaydedici birimleri kullanırken yüksek anahtarlama etkinliğini, sinyal geçişlerinde yüksek güç kaybıyla gerçekleşmiştir. Bu kaybı önlemek amacıyla Hamming uzaklık algoritmasıyla filtrenin gerçekleşmesi esnasında anahtarlamanın etkinlik ölçeği biçimlendirilmiştir. Böylece hamming uzaklık algoritmasıyla minimizasyonu sağlanır ve optimizasyon tekniğiyle de yani genetik algoritmayla sinyal geçişi azaltılmıştır ve güç kaybı da azaltılmıştır [5].

2009 yılında KAÇAR ve arkadaşları tarafından kablosuz algılayıcı ağlar için genel amaçlı, modüler, uygulamalara bağlı olarak ve uzaktan izlenebilmesi için bir web tabanlı bir çalışma yapılmıştır.ASP .NET tabanını kullanarak MATLAB Builder NE ve MATLAB Web figure araçları kullanılarak çalışma gerçekleştirilmiştir. Bua arayüzde kablosuz algılayıcılarından gelen veriler gerçek zamanlı olarak görülebilmektedir ve elde edilen veriler her düğüm için ve karşılaştırmalı olarak grafikler elde edilebilir [6].

2009 yılında Kaya ve arkadaşları tarafından yapılan bu çalışma da analog süzgeçlere uygulanan bilineer dönüşümü, genetik algoritma kullanılarak sonsuz dürtü yanıtlı (IIR) sayısal süzgeç tasarlanmıştır. Süzgeç giriş-çıkışını gösteren fark denklemi katsayıların istenen şekilde üretilmesi gereklidir. Bu çalışmada genetik algoritmayla sayısal süzgeç tasarlayabilmek için analog süzgece uygulanan dönüşüm katsayıları kullanılmıştır. Süzgeç katsayılarının optimizasyonu gerçekleştirdikten sonra elde edilen sonuçlarla genlik cevapları çizdirilmiş ve arzu edilen çizimle karşılaştırılmıştır. Genetik algoritmanın etkileri çalışma üzerinde incelenmiştir [7].

2009 yılında ÇANKAYA ve arkadaşları tarafından yapılan çalışmada matlabın elektronik devrelere uyarlanabilme özelliği kullanılarak RLC devrelerinin eğitimine yönelik olarak MATLAB GUI programı kullanılmıştır. Filtre çeşitlerinden alçak geçiren, yüksek geçiren, band geçiren ve band durduran tasarlanmış ve frekans cevapları, kök yer eğrisi gibi yöntemlerle gerçekleştirilmiştir [8].

2008 yılında Sabbir U. AHMAD tarafından gerçekleştirilen doktora tez çalışmasında, sayısal filtrelerle ilgili olarak dört ayrı çalışma gerçekleştirilmiştir. İlk çalışma olarak GA tabanlı kesir gecikmeli filtrelerin tasarımı geliştirilmiştir. Bu yaklaşımlar FARROW yapısı olarak adlandırılan alçak geçiren IIR filtrelerin ve FD FIR filtre katsayılarının belirlemek için genel arama tekniğinin faydalarından yararlanılmıştır. GA ile gerçekleştirilen genlik cevapları ve gecikme karakteristikleri aynı zamanda en küçük kareleme yöntemiyle de gerçekleştirilerek, karşılaştırılmıştır. İkinci çalışma olarak, gecikme denklemlerinin tasarımı için genetik algoritma kullanılmıştır. Denkleştirici katsayılar band geçiren filtrelerde kullanılarak, grup gecikmelerinde amaç fonksiyonu olarak kullanılmıştır. Üçüncü çalışma olarak, GA yaklaşımı daha az çoğaltıcılı FIR filtrelerin tasarımında kullanılmıştır. GA deneysel tasarımı tekniği olduğu için ortogonal genetik algoritma tekniği geliştirilmiştir. Bu teknikle de doğrusal fazlı FIR filtrelerde sabit noktalı gerçekleşimler kullanılmıştır. Dördüncü çalışma olarak, asimetrik FIR filtrelerin tasarımı GA kullanılarak geliştirilmiştir. Band geçiren filtrenin genlik cevabı ve grup gecikmelerindeki kararlılık incelenmiştir. Bu yaklaşımın yanında ek olarak WLS tasarım methoduyla gerçekleştirilerek, karşılaştırılma sağlanmıştır. Son çalışma olarak da GA ile birlikte yarı Newton algoritması birleştirilerek IIR filtrelerin tasarlanmıştır. Hibrid algoritma da esneklik, güvenirlik ve kesinlik özellikleri birleşerek istenen genlik cevapları kolay bir şekilde gerçekleştirilmiştir. Geliştirilen algoritmalar ile ortaya çıkan genlik cevapları ve grup gecikmeleri grafiksel olarak da görülmektedir[9].

2008 yılında Kaya ve arkadaşları tarafından gerçekleştirilen çalışma da genetik algoritma kullanarak aktif filtre tasarlanmıştır. İstenen özellikleri verecek filtre transfer fonksiyonu hesaplamasında klasik optimizasyon ve genetik algoritma yöntemi kullanılmıştır. Klasik optimizasyonda filtrenin transfer fonksiyonunu hesaplarken karmaşıklık, işlem basamak fazlalılığı ve harcanan zaman artmaktadır. Bu sebepten dolayı eleman değerlerinin eşit seçilmesi tasarımcıya farklı eleman değerlerinin verilememesi kullanıcıyı sınırlandırmıştır. Genetik algoritmayla analog filtre çeşitlerinden Sallen Key filtresinin eleman değerleri hesaplanmıştır. Bu hesaplanan eleman değerlerin farklı seçilmesi tek bir eleman değerine bağımlılığı ortadan kaldırarak istenen özelliklerde transfer fonksiyonu elde etmeyi başarmıştır. Sonuç olarak genetik algoritma sayesinde MATLAB programında yapılan tasarımda

hesaplama zorlukları ve karmaşıklıkları ortadan kalkmıştır. Band geçiren filtrenin genlik cevapları istenen ve hesaplanan değerleri gösterir şekilde karşılaştırılmıştır [10].

2008 yılında AGNIHOTRI tarafından gerçekleştirilen tez çalışmasında genetik optimizasyon algoritmasıyla hamming uzaklık minimizasyonun birleşimden oluşan algoritmayla MATLAB programında dört farklı FIR(sonlu dürtü cevabı) filtre üzerinde gerçekleştirilmiştir. Bu filtrelere pencereleme metotları da eklenmiştir. Bulunan katsayılar iki tabanında 16 bitlik olarak nicelenmiştir. Tasarımın sonucu olarak ideal ve istenen frekans cevaplarının simülasyonu gerçekleştirilmiştir. Geliştirilen bu teknikle günümüzdeki iletişim sistemlerinin de kullanılan sayısal sinyal işlemcisi çeşitlerinden FIR (sonlu dürtü cevabı) filtreleri kullanılır ve yukarıda anlatılan teknik uygulanmıştır. Bu teknikle de düşük güç kaybı ve bataryanın ömrünü maksimum kullanabilecek şekilde sonuç elde edilmiştir. Bu çalışmanın genetik algoritma yerine bulanık mantık ve yapay sinir ağlarıyla da hamming uzaklık minimizayonu birleştirerek gerçekleştirilebileceği öne sürülmüştür [11].

2008 yılında Kaya ve arkadaşları tarafından yapılan bu çalışma da butterworth filtre tasarımı için gerekli olan transfer fonksiyonun kutup değerleri genetik algoritma kullanılarak elde edilmiştir. Filtre çeşitlerinden alçak geçiren ve band geçiren filtreleri kullanılmıştır. Teorik olarak yapılan elle hesaplama ve nümerik olarak hazır komutlar yardımıyla bulunan filtre katsayı değerleri aynı olup bu değerler ile GA kullanılarak geliştirilen programın çalışmasından sonra en iyi kromozom değerine ait katsayı değerleri filtre transfer fonksiyonun pay ve payda katsayı değerlerini temsil etmektedir. İstenen genlik cevabı ve arzu edilen genlik cevapları karşılaştırılmıştır. Genetik algoritmanın teorik olarak elde edilen filtrelerin dereceleri arttıkça işlem karmaşıklığı artmıştır oysa genetik algoritmada filtre derecesi arttıkça hesaplama zorluğu ortadan kalkmıştır[12].

2008 yılında Kaya ve arkadaşları tarafından yapılan bu çalışmada genetik algoritmayla sayısal filtre katsayıları hesaplanmıştır. Sayısal filtre çeşitlerinden de sonlu dürtü cevabı veren FIR filtre katsayılarının optimizasyonu gerçekleştirilmiştir. Filtre katsayıları hesaplanırken GA'nın kodlama yöntemlerinden değer kodlama

yöntemi, çaprazlama ve mutasyon oranını farklı değerlerde uygulayarak başlangıç popülasyonundaki çeşitliliği artırmıştır. Düşük dereceye sahip filtrelerin genlik cevabına ulaşılmaktadır. MATLAB'ın kendi içinde komutlarla elde edilen sonuçlar ile elde edilen sonuçları karşılaştırılmıştır ve grafiksel olarak sonuç elde edilmiştir [13].

2006 yılında Barros ve arkadaşları tarafından geliştirilen çalışmada basit bir FIR filtre aracı geliştirilmiştir. Tasarlanan araç genetik algoritma kullanılarak geliştirilmiştir. Tasarlanan aracın diğer araçlardan farklı olarak iki özelliği vardır. Ara yüzde genetik algoritma parametrelerinden minimum sayıda ihtiyaç vardır bunlarda popülasyon büyüklüğü, nesil sayısıdır ve FIR filtre tanımlamalarını, fazladan iki parametre eklenir. Arayüz sayesinde işlem karmaşıklığı saklanmıştır. GA ile elde edilen katsayılar kullanılarak elde edilmiş olan frekans cevabı Parks-Mcclellan metoduyla karşılaştırılmıştır, genetik algoritmanın ideal cevaba daha yakın cevaplar verdiği gösterilmiştir[14].

2006 yılında Turgay KAYA tarafından yapılan yüksek lisans tez çalışmasında Genetik Algoritma (GA) kullanılarak sayısal filtre çeşitlerinden FIR (Sonlu Dürtü Cevabı) filtre katsayılarının optimizasyonu gerçekleştirilmiştir. Bu optimizasyonu gerçekleştirmek için GAFKAT adlı bir yazılım geliştirilmiştir. Yazılımda GA parametrelerinden çaprazlama ve mutasyon parametrelerini kullanarak gerçekleştirilen popülasyondan uygunluk değerleri en iyi olan katsayılar seçilerek ideal sonlu dürtü cevabının genlik cevapları çizdirilmiştir ve daha önce gerçekleştirilen çalışmalar ile karşılaştırılarak GA'nın FIR filtreler üzerinde istenilen sonuçları verdiği gösterilmiştir[15].

2005 yılında Tzeng tarafından yapılan çalışmada genetik algoritma yaklaşımıyla tek boyutlu doğrusal fazlı kompleks FIR sayısal filtresi gerçekleştirilmiştir. Analitik ve kompleks işlem uygulamalarında kompleks katsayılarla gerçekleşen sayısal filtreleri içerir ve tek-kenar band modülasyon sistemlerine uygulanır. Trigonometrik polinom için karmaşık filtrelerin genlik cevabı, hata ve uygunluk fonksiyonu en küçük kareler durumunda formüle edilmiştir. Tasarlanan kompleks

filtre de GA yaklaşımı başarılı bir şekilde uygulanmıştır. Birkaç sayısal tasarım örnekleriyle gerçekleştirilen tasarımın etkinliği ve kapasite gösterilmiştir[16].

2004 yılında Karaboğa ve arkadaşlarının yaptıkları çalışmada Genetik Algoritma kullanılarak sayısal filtre tasarlanmıştır bu tasarlanma da minimum faz elde edilmeye çalışılmıştır. Genetik algoritma ile optimizasyonu sağlanmış ideal genlik cevabı elde edilirken band geçiren ve band durdurma bölgelerinde MSE hata fonksiyonu, band geçirme bölgesinde ise MAE hata fonksiyonu kullanılmıştır. GA ile birlikte kullanılmış olan hata fonksiyonlarıyla elde edilmiş genlik cevaplarını grafiksel olarak gösterilmiştir. Simülasyon sonuçlarında görüldüğü gibi minimum faz sağlanmıştır ve band geçiren ve band durduran bölgelerde zayıflamalar azalmıştır[17].

2004 yılında Tzeng tarafından gerçekleştirilen bu çalışmada iki boyutlu simetrik özelliklerle iki boyutlu tasarlanan FIR (sonlu dürtü cevabı) sayısal filtre için genetik algoritma yaklaşımı kullanılmıştır. Doğrusal fazlı filtre tasarımlarının geniş çeşitliliği sayesinde iki boyutlu doğrusal fazlı FIR sayısal filtrelerinin simetrik özellikleri ortaya çıkmıştır. Buradan da iki boyutlu simetrik/asimetrik dizilerin ikisinde de 16 filtre çeşidi vardır ve filtre uzunlukları tek veya çift olarak gerçekleştirilmiştir. Genlik cevaplarına uygun gelen çeşitler tablolaştırılmıştır. Ayrıca karesel düzlem, yarım düzlem ve tam düzlem filtreleri genetik algoritma yaklaşımıyla sayısal örneklerle tanımlanmıştır. Görsel olarak gerçek genlik cevapları ve istenen genlik cevapları verilmiştir[18].

2003 yılında Liang ve arkadaşları tarafından yapılan çalışmada genetik algoritma kullanılarak doğal işaretli rakamlı (CSD-canonical signed-digit) katsayılar kullanarak 1-D IIR filtreler tasarlanmıştır. CSD sayılarıyla yenilenmiş teknik de çaprazlama ve mutasyon işlemlerinden geçen geçersiz katsayılar iyileştirmek için popülasyona geri döndürülür. Burada en iyi bireyler seçilerek CSDli formatta katsayılar hesaplanmıştır. Bu katsayılarla kutuplar, genlik(db) cevabı ve grup gecikme simulasyonları gerçekleştirilmiştir. Algoritmanın uygunluğu alçak geçiren IIR filtre üzerinde hesaplamalar yapılarak gösterilmiştir[19].

1998 yılında Redmil ve arkadaşı tarafından düşük karmaşıklı sonlu dürtü cevabı(FIR) filtre tasarlanmıştır. Genetik algoritmayı buluşsal grafik tasarım algoritmalarıyla birleştirmiştir. Performans, karmaşıklık ve filtre dereceleri farklılaştırarak çözümler geliştirilmiştir. Örnek sonuçlar incelendiğinde tek ve çift boyutlu filtreler için gerçekleştirildiğinde yüksek performans ve karmaşıklık çıkmıştır. Karmaşıklık filtreleri sayısal katsayılarla sınırlandırarak kısıtlamıştır. Genetik algoritmanın kodlama türlerinde sayısal değerler rahatlıkla kullanıldığından algoritmanın işlevselliği artmıştır. Bu katsayıları etkili olacak şekilde ilkel operatör yönlendirici grafik (PODG) kullanmıştır. PDOG'nin kullanımı doğal işaretli sayı(CSD) veya ikinin işaretli gücü gerçekleşimleri önemli düzeltmeleri sağlamıştır[20].

1995 yılında Arslan ve arkadaşları tarafından yapılan bu çalışma da genetik algoritma ile analog ve dijital filtre tasarlanmıştır. Bu tasarımın ortaya çıkmasının nedenleri arasında geleneksel yaklaşım ile standart çok terimli transfer fonksiyonlardan biri seçilmiştir. Bunun sonucunda cevap belirlenimleri elde edilir. Bu transfer fonksiyon standart devre yapılarında kullanılır ve gerçekleştirilen yaklaşım yetersiz kalmıştır ve optimizasyon yaklaşımı gerekmiştir. İhtiyacın sonucunda standartsız tepki cevapları ve sayısal filtreler için hesaplama karmaşıklığını ve sonlu aritmetik sonuçlar gibi uygun sonuçlar elde edilmesi göz önünde bulundurulmuştur ve genetik algoritmayla sonlu kelime uzunluklu IIR sayısal filtrelerin tek adım tasarımına uygulanmıştır. GA ile IIR filtrelerin katsayıları, üretilen popülasyondan çaprazlama ve mutasyon işlemleri uygulandıktan sonra en ideale yakın olanları seçilmiştir. Genlik ve faz cevaplarının algoritmayla elde edilen gerçekleşimleri simülasyon olarak verilmiştir[21].

1994 yılında Nurhan KARABOĞA tarafında yapılan doktora tez çalışmasında sayısal filtre çeşitlerinden FIR (sonu dürtü cevabı) katsayıları GENETİK ALGORİTMA (GA) kullanılarak kuantalama tekniği geliştirilmiştir. Kuantalanmış sayılarla tasarlanan filtre karakteristiğiyle orijinal filtre karakteristikleri farklı olmaktadır. Bu yüzden bu çalışmada da sabit noktalı sayılar ve kayan noktalı sayılar kullanılarak elde edilen katsayıları GA kullanılarak yuvarlaştırılmıştır. Yuvarlaştırmanın amacı ise katsayılarda kullanılan sayılarının bit uzunluğunun

mümkün olduğunca az olması istenmektedir. Ayrıca GA ile karşılaştırma yapmak amacıyla katsayılar el ile yuvarlatılması da gerçekleştirilmiş. Elde edilen katsayılar ile frekans cevaplarının simülasyonları gerçekleştirilmiştir. Bu grafikler karşılaştırılarak GA ile elde edile sonuçlarda filtre cevabın durdurma kazancında bozulmanın gerçekleşmediği görülmektedir. Sonuç olarak GA ile tasarlan filtre de daha az bit kullanılarak elde edilen katsayılar sayesinde donanım tasarlanırken maliyetinin düşürülmesi sağlanmıştır ve gerçek zamanda çalışması sırasında ise zaman açısında tasarruf sağlanacaktır[22].

1991 yılında Suckley ve arkadaşları tarafından FIR(sonlu dürtü cevabı) filtre tasarım metodu otomatik, hızlı ve en az hesaplama karmaşıklığıyla tasarlanmıştır. Hesaplama karmaşıklığı minimizasyonu sağlanacak olan bütün filtre çeşitlerinde zorunlu bir ihtiyaçtır. Bu sorunu en aza indirgemek için filtre sentezleme de genetik algoritma kullanılmıştır. Genetik algoritma kodlama yöntemlerinde binary kodlama kullanılmıştır. Her jenerasyon çalıştırıldıktan sonra katsayılar ideale yakın cevap verildiği için en uygun bireyler popülasyondan seçilmiştir. Genetik algoritmanın performansı ardışıl algoritmalarla elde edilen sonuçlar ile karşılaştırılmıştır. Filtre çeşitlerinden sadece alçak geçiren filtre üzerinde frekans cevabı elde edilmiş ve simülasyonu gerçekleştirilmiştir. Bu çalışmanın yüksek geçiren, band geçiren ve band durduran filtreler üzerinde geliştirilebileceği öngörülmüştür[23].

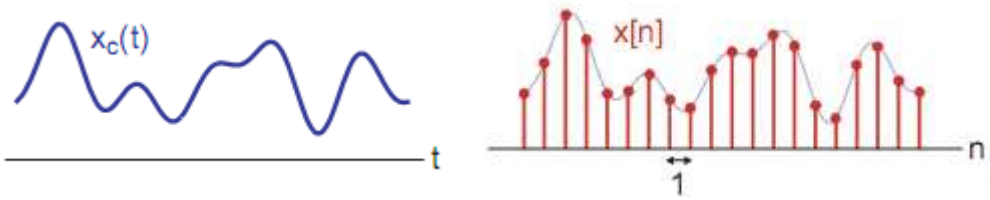
Bu çalışmada da, Genetik Algoritma kullanılarak elde edilen FIR filtre katsayılarının hesaplanmasında ve elde edilen katsayı değerler ile filtre cevaplarının grafiksel olarak elde edilmesine yönelik iki farklı arayüz tasarlanmıştır. Bu arayüzlerden biri MATLAB GUI ile diğeri ise ASP.NET ile hazırlanmıştır. İki farklı arayüzden elde edilen sonuçlar karşılaştırılmıştır.

BÖLÜM 2. SAYISAL FİLTRELER

2.1. Giriş

Sinyal fiziksel bir sistemin durumu hakkında bilgi taşıyan ve bir veya birden fazla değişkene bağlı olan fonksiyonlardır. Sinyallere örnek olarak elektriksel sinyaller, akustik sinyaller (konuşma, müzik), biyolojik sinyaller (gendeki baz serileri) gibi verilebilir.

Sinyaller ayrımı zamana göre gerçekleştirilebilir. Bunlar sürekli zamanlı sinyal, ayrık zamanlı sinyaldir.



Şekil 2.1. Sürekli ve ayrık zamanlı sinyal örnekleri [33]

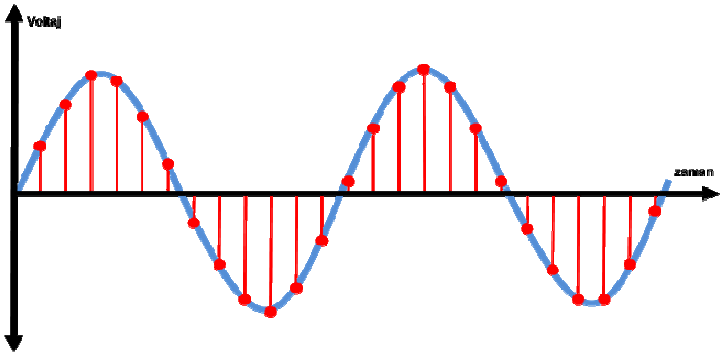
2.1.1. Analog sinyal

Analog sinyal “sürekli” olarak da adlandırılır. Genliği ve zamanı minimum ve maksimum değerler arasında herhangi bir değeri alabilir. Analog sinyal, sinyal bilgilerini iletmek için bazı orta özellikleri kullanır. Örneğin bir kadranlı baramotre basınç bilgilerini sinyal olarak aktarmak için dönerek çalışan pozisyonu kullanır, elektrikle ilgili olarak voltaj, frekans, akım ve şarj gibi genel olarak kullanılan özellikleridir.

Analog sinyal teorik olarak sonsuz çözüme sahiptir. Pratikte ise analog sinyal gürültü ve sınırlı yetişme hızına bağlıdır. Bundan dolayı, analog ve sayısal sistemler çözüm ve bant genişliğinin sınırlanırına bağlıdır.

2.1.2. Sayısal sinyal

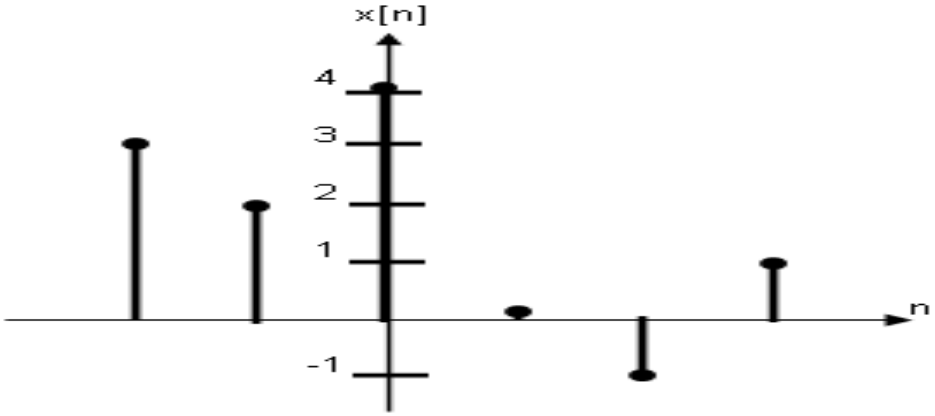
Sayısal sinyal ayrık zamanlı işaretlerde kullanılır. Ayrık zamanlı işaretler genliği sadece ayrık değerler alabiliyorsa veya ayrık sayıların farklı seviyeleri olarak ifade edebiliriz. Örneğin örneklendirilmiş ve sayısallaştırılmış analog sinyal veya sayısal sistemde sürekli dalga biçimi(waveform) sinyallerini ve bit akımını(bit-stream) temsil eder. Şekil 2.2'de analog sinyalin sayısal olarak işaretlenmesi gösterilmektedir.



Şekil 2.2. Analog sinyalin sayısal olarak işaretlenmesi

2.1.2. Ayrık zamanlı işaretler

Ayrık zamanlı işaretler bir dizi sayısaldan oluşturulur ve dizinin sayıları x_n, x[n], x[nT] şekilde ifade edilir. Bu gösterimde n bir tamsayıdır ve dizide yer alan sayıların sıra indisini gösterir. x[nT] biçiminde, sürekli zamanlı x(t) işaretinin t=nT olduğu zamanlardaki örneklenmesinden elde edilebilir.(Şekil 2.3) Ayrık zamanlı bir dizi açık olarak ifade edildiğinde dizinin elemanlarından altına yerleştirilen ok orjin(n=0) noktasını gösterir.



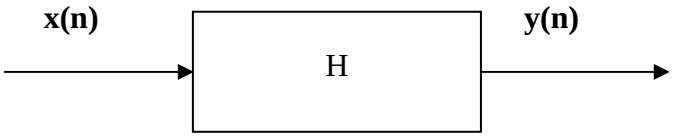
x[n] = {3 2 4 0 -1 1 }
↑
n=0

Şekil 2.3. Ayrık zamanlı belirtilen işaret dizisinin grafiksel olarak gösterilmesi

2.1.2. Ayrık-zamanlı sistemler ve özellikleri

Ayrık zamanlı sistem girişe verilen x(n) giriş işaretini bir çıkış işaretine y(n) dönüştüren bir sistemdir. (Şekil 2.4) Giriş işareti x(n), sistem tarafından y(n) dönüştürülmektedir. Bu dönüşüm y(n)=H.x(n) dönüşüm kuralı kullanılarak gerçekleştirilebilir. [1]

Eğer sonlu sayıda genlik değerlerine sahip olan x(n) ve y(n)'li ayrık zamanlı sistemler sayısal sistemlerdir.



Şekil 2.4. x(n) girişli ve y(n) çıkışlı ayrık zamanlı sistem

[1] H burada sistemin yerine kullanılmıştır.

Sayısal işaret işlemede kullanılan sistemler doğrusal, zamanla-değişmeyen, nedensel ve kararlıdır.

— **Doğrusallık:** Sayısal bir sistemin doğrusal olabilmesi için çarpımsallık ve toplamsallık özelliklerini sağlaması gerekir.

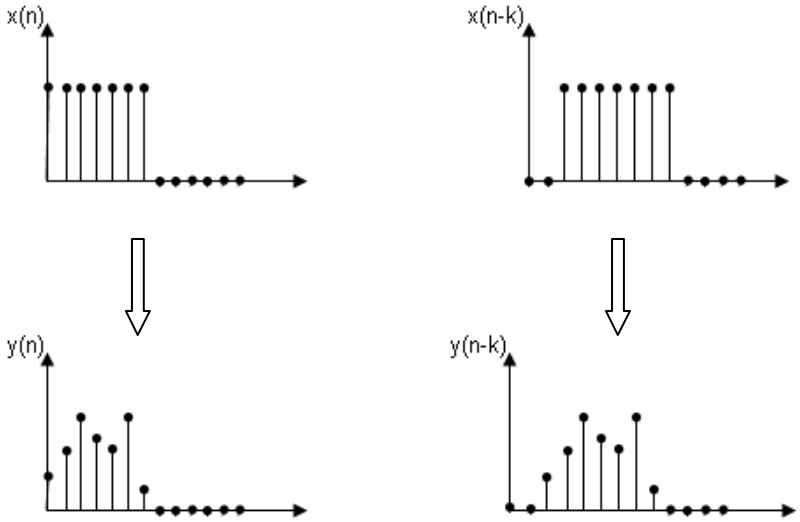
$$y_1(n)=H.\ x_1(n)$$
$$y_2(n)=H.\ x_2(n) \qquad (2.1)$$

Bir sistemin doğrusal olabilmesi için aşağıda bulunan denklemi sabit olan a ve b sayılarını kullanılır.

$$\begin{aligned} H[a.x_1(n)+ b.x_1(n)] &= H[a.x_1(n)]+ H[b.x_1(n)] \\ &= a.H[x_1(n)]+ b.H[x_1(n)] \\ &= a.y_1(n)+ b.y_2(n) \end{aligned} \qquad (2.2)$$

koşulunu sağlayan sistemler doğrusal(lineer) sağlamayanlar ise doğrusal olmayan(non-lineer) sistemlerdir.

— **Zamanla Değişmezlik:** Sayısal bir sistemin giriş ve çıkış arasındaki ilişki zamanla değişmiyorsa bu sistemlerin zamanla değişmezlik özelliğine sahiptir. Bu sistemde girişe uygulanan x dizisini aynı zamanda bağımsız olarak çıkışta y dizisi olarak üretilebilir. Sistemin zamanla değişmez olması için her x(n) girişi için [x(n-k)]=y(n-k) sağlanmalıdır.



Şekil 2.5. a) Sistemin x(n) girişine cevabı b) Sistemin x(n-k) girişine cevabı

Yukarıdaki Şekil 2.5'te sistemin x(n) ve x(n-k) girişlerine uygulanan zaman farkının y(n) ve y(n-k) çıkışlarında da eş zamanlı olarak zamanda değişmezlik görülmektedir.

- **Nedensellik:** Sistemin çıkışı sadece girişe uygulanan ve geçmişteki girişlere bağlıysa yani gelecekte girişlere bağlı olmamasıdır.
- **Kararlılık:** Girişe uygulanan sınırlı giriş dizisinin çıkışta da sınırlı bir çıkış dizisi üretebiliyorsa bu sistemlere kararlı sistemler denir.

2.2. Sayısal Filtreler

Sayısal filtreler sayısallaştırılmış analog sinyalleri kullanarak çalışmaktadır. Sayısallaştırma, analog sinyalin örneklenmesini gerektirir ve sayısal değere dönüştürülür. Örneğin bir analog sinyal örneklendikten sonra genliğine bağlı olarak da sayısal değere dönüştürülmesi gerekir. Sayısal filtrenin temelinde veri örnekleme, sayısallaştırmayı(eksik örnekleme, aşırı örnekleme, ara değerleme ve örnek seyrelteme) işlemleri bulunmaktadır.

- **Analog – Sayısal Dönüşümü**

Analog verilerin kullanılabilmesi için ilk aşama olarak sayısal veri hale dönüştürülmesi gerekir. Analog sinyal örnekleri belirli periyot aralıklarında alınmakta ve daha sonra sayısal şekle dönüştürülmektedir. Dönüştürülen sayısal şekli giriş geriliminin örnekleme anındaki verinin ikilik gösterimidir. Genel olarak analog-sayısal dönüştürücüler 8 veya 16 bit genişliğinde olan bir veri sözcüğüdür.

— **Eksik Örnekleme**

Örnekleme frekansının analog sinyalinin en yüksek frekansından iki kat daha az olması durumunda eksik örnekleme görülür. Eksik örnekleme, istenilen sinyallerin banttan geçemeyen örtüşen sinyallerden oluşmaktadır. Örnekleme işleminde, zaman bölgesinde örnekleme palsı ile analog sinyal çarpılmaktadır. Bu işlemin çıkışında sonuç olan frekans aralığı örnekleme frekansı ile analog sinyalin toplamı ve farkıdır. Örneğin 5 kHz'lik istenmeyen bir sinyal ile 7 kHz'lik bir örnekleme frekansı işleme tabi tutulduğunda çıkış sinyali (7+5) kHz ve (7-5) kHz veya 12 kHz ile 2 kHz'yi içeren bir frekans aralığına sahip olacaktır.

Eksik örneklemede istenmeyen sinyallerin bantta çarpışmamasına dikkat edilmelidir. Örnekleme frekansı analog sinyalin bant genişliğinin iki katından fazla olmalıdır. Ancak örnekleme frekansı, analog sinyalin orta frekansından az olabilmektedir. Aralık işaretlerinin yerleri denklem 2.3'te gösterilmektedir.

$$f_1 = | n. f_s \pm f_c | \qquad (2.3)$$

Denklem 2.3' de görülen f_1 işaretin frekansı, f_s örnekleme frekansı, f_c analog sinyalin frekansıdır. İşaret, aralıkta tekrarlanmasını çarpan tamsayısı olan n ile gerçekleştirir.

Sinyallerin üst üste gelmesini önlemek amacıyla denklem 2.4'de gösterilen formüllerle gerçekleşir.

$$f_1 = | n. f_s \pm f_c |$$

$$f_1 + B + \frac{\omega - B}{2} < (f_s - f_1) \quad (2.4)$$

Şartını sağlaması gereklidir. Bu denklemedeki B bant genişliği, ω ise geçen bant ile duran bant arasındaki frekans aralığının en az olmasıdır.

— **Aşırı Örnekleme**

Analog sinyal frekansının en yüksek değerinin birkaç katı hızdaki örneklemeye aşırı örneklem denir. Aşırı örneklemenin avantajı olarak sinyallerdeki bozulmaları azaltır, dezavantajı olarak ise sinyallerin diğer görevlerini yerine getirmemesi anlamına gelir. Bu eksikliği gidermek için ise hızlı bir şekilde veri işlemesi gerekir.

— **Örnek Seyreltme**

Örnek seyreltme işlenen veri hızını indirgemek amacıyla kullanılır. Sinyal kendisinden fazla olan yüksek frekanstaki istenmeyen sinyallerin ortama alınmasını sağlanamadığı durumlarda kullanılır. Örnek seyreltme işleminde sinyaller örnekleme hızları yüksek olacak şekilde örneklenir ve sayısallaştırılmış sinyal örnekleme hızını istenen sinyale uygun değere getirerek azaltabilir.

2.2.1. Sayısal filtrelerin avantajları ve dezavantajları

Sayısal filtreler analog filtrelerle karşılaştırılmış ve elde edilen avantajlar aşağıda verilmiştir.

1- Sayısal Filtreler genelde (DSP) kullanılarak tasarlandıkları için, filtre karakteristikleri entegre programlanarak filtre tasarlanır, bu yüzden filtrede bir değişiklik yapılmak istendiğinde sadece yazılımı değiştirmek yeterli olmaktadır, oysa analog filtrelerde filtre devresini değiştirmek gerekir bu da maliyet bakımından ve zaman bakımından bakıldığında sayısal filtrelerin üstünlüğü açıkça görülmekte.

2- Sayısal filtreler kolayca tasarlanıp bilgisayar simülasyonlarında veya başka ortamlarda kolayca çalıştırılıp test edilebilir analog filtrelerde bu mümkün değildir.

3- Analog filtrelerde filtrenin çalışma noktası ve kararlılığı ortam sıcaklığına bağlıdır. Ortam sıcaklığındaki artışlar filtrenin çalışmasını doğrudan etkileyeceği için analog filtreden tam verim alınamaz. Oysa sayısal filtreler sıcaklık değişimlerine karşı daha kararlı yapıdadırlar.

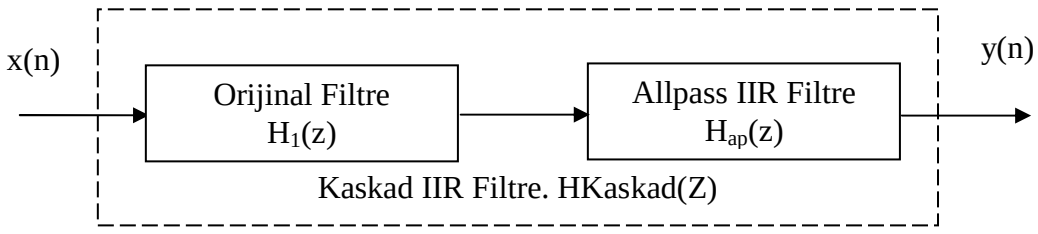
4- Analog filtrelerden ayrı olarak, sayısal filtreler düşük frekanslı sinyallerde analog filtrelere göre daha az hata oranı ile çalışır. DSP Teknolojisinin gelişmesiyle günümüzde sayısal filtreler yüksek frekansta çalışan RF uygulamalarında kullanılmaktadır.

5- Sayısal filtreler sinyal işleme yönünden analog filtrelere göre daha fazla yönteme sahiptirler.

Dezavantajları olarak enerjiyi geçirmemek, örnekleme etkileri, güç kaynağı gerektirmekte de ve frekans aralığı sınırlamaları mevcuttur.

2.2.2. Filtrelerde kullanılan terimler

— **Allpass filtre:** Bu terim, IIR Filtre için cevap büyüklüğünün filtrenin tüm frekans alanından daha fazla olduğu ve faz cevabının kararsız olduğu IIR Filtreler için kullanılır. Bu tip Filtreler tipik olarak bir IIR Filtrenin sonuna kaskad olarak bağlanır. Şekil 2.7'de bu tip filtrenin kullanımı gösterilmektedir.



Şekil 2.6. Kaskad Filtrelerin Kullanımı

Allpass Filtrenin H_{ap} (Z) faz cevabı, orijinal IIR filtrenin nonlineer fazına cevap verecek şekilde tasarlanabilir. Bu kaskad bağlama sonucunda oluşan H_{Kaskad} (Z) , H_1(Z) ye göre daha lineer bir davranış gösterir, bu da iletişimde istenen bir durumdur.

— **Zayıflama (Attenuation):** Genişleme zayıflamasıdır, genelde birimi dB olarak verilir. Sinyalin filtre de işleme tabi tutulduğu zaman zayıflama miktarıdır. Bu oran, sinyalin filtre çıkışındaki genliğinin girişindeki genliğine oranıdır.

$$\text{Zayıflama} = 20.\log_{10}\left(\frac{a_{out}}{a_{in}}\right)db$$

Bu oran a_{out} değerinin a_{in} den küçük olduğu değerler için verilmiştir. Ve bu durumda zayıflama (attenuation) oranı negatif olur.

— **Band Geçiren Filtre:** Bu tanım tek frekans bandını engelleyen, diğer frekans bandlarını geçiren filtre için kullanılır.

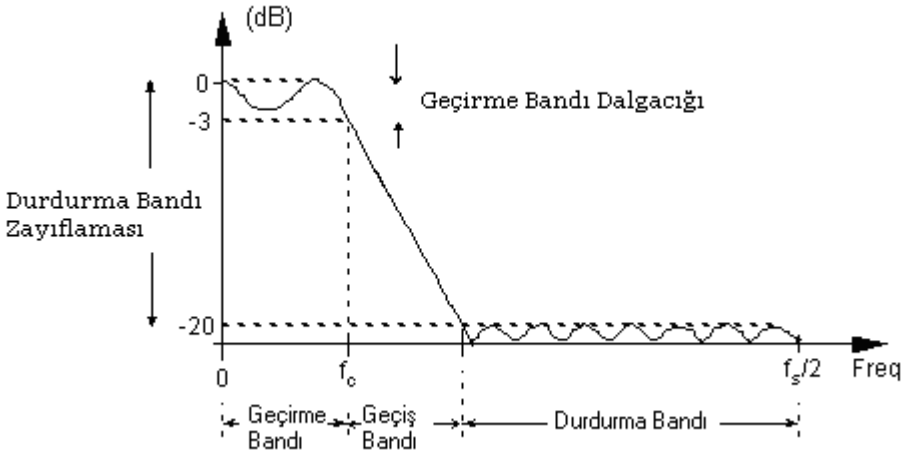
— **Band Genişliği (Bandwidth):** Sinyal işleme terminolojilerinde bu kelime için birçok tanım mevcuttur. Biz bu terimi, geçirme bandının frekans genişliği olarak tanımlayacağız. Bir alçak geçiren filtre için band genişliği (Bandwidth) filtrenin kesim frekansına (Cutoff frequency) eşittir. Bir band geçiren filtre (Bandpass filtre) için band genişliği -3db noktasının sınırladığı frekans genişliği olarak tanımlanır.

— **Kesim Frekansı (Cutoff Frequency):** Alçak geçiren filtre için kesim frekansı, üst geçirme bandı frekansına, yüksek geçiren filtreler için kesim frekansı, alt geçirme bandı frekansına eşittir. Aşağıdaki Şekil 2.7’de bir alçak geçiren filtrenin kesim frekansını göstermektedir[27].

— **Geçirme Bandı (Passband)** – Filtrenin, sinyal enerjisini geçirdiği frekans bandıdır.

— **Geçirme Bandı Dalgacığı (Passband Ripple**)– Şekil 2.8'de gösterildiği gibi geçirme bandındaki istenmeyen dalgalanmalar olarak tanımlanır.

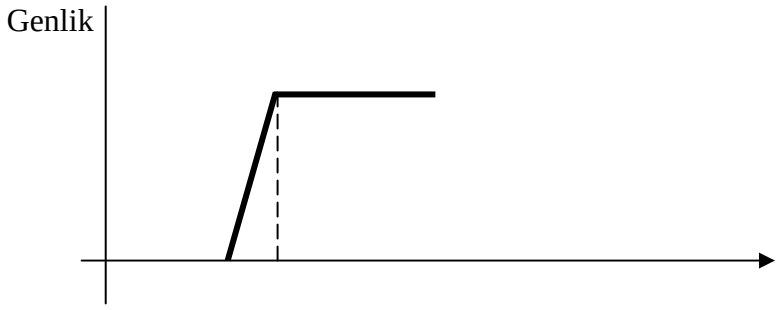
— **Yükselme Zamanı (Roll-off)** – Bu terim filtre cevabının, geçirme bandı ile durdurma bandı arasında olan geçiş bölgesinde filtre cevabının eğimini göstermektedir.



Şekil 2.7. Alçak geçiren bir filtrenin frekans cevabı

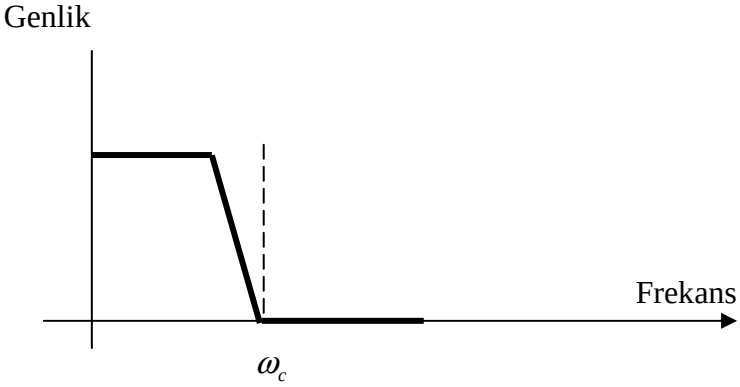
— **Decibels(dB):** Bu değeri filtre tasarımlarında kesim frekansını ve durdurma bandını göstermek için kullanılır. Şekil 2.7'de alçak geçiren filtre örneğinde gösterildiği gibi filtrenin kesim frekansı(-3dB) , durdurma bandı seviyesi de (-20dB) dir.

— **Yüksek Geçiren Filtre(Highpass Filter):** Şekil 2.8'de görüldüğü gibi, sadece yüksek frekansları geçirip alçak frekans bileşenlerini durduran bir filtredir.



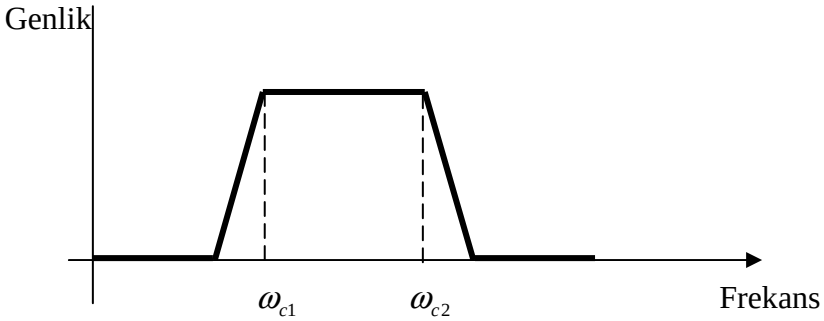
Şekil 2.8. Yüksek geçiren filtre

— **Alçak Geçiren Filtre(Low-pass Filter):** Şekil 2.9'da görüldüğü gibi alçak frekansları geçirip, yüksek frekansları geçirmeyen filtredir.



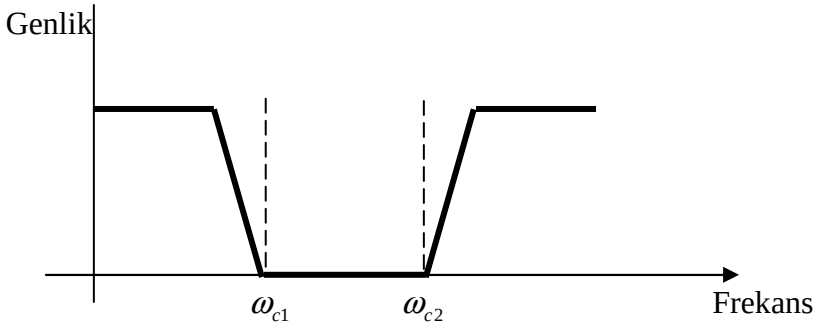
Şekil 2.9. Alçak geçiren filtre

— **Band Geçiren Filtre (Bandpass Filter):** Alt(ω_{c1}) ve üst(ω_{c2}) kesim noktaları arasında kalan Şekil 2.10'da gösterildiği gibi sadece tek frekans bandını geçirip, diğer frekans bantlarını durdurur[28].



Şekil 2.10. Band geçiren filtre

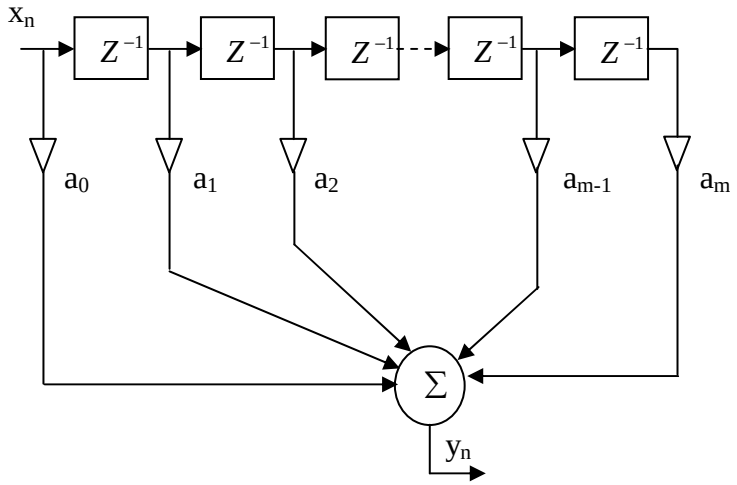
— **Band Durduran Filtre (Bandstop Filter):** Alt(ω_{c1}) ve üst(ω_{c2}) kesim noktaları arasında kalan frekans bandını durdurur[26].



Şekil 2.11. Band Durduran filtre

2.3. Sayısal Sonlu Dürtü Yanıt (FIR) Filtre Tasarımı

Sonlu dürtü yanıt (FIR) filtresi yapısı genel olarak Şekil 2.12'de gibidir. Şekilde gösterilen z^{-1} ile örnekler arasında, bir örnek gecikme sağlanmaktadır. Bu şekilde birbirine kaskat bağlanmış m adet bölüm görülmekte. Filtrenin yükü şimdiki m ve bir önceki m değerlerinin toplamından oluşmaktadır. Bu değerlerin toplamı filtrenin çıkışını oluşturmaktadır. Filtre katsayıları (Filtre yükü) merkezdeki değere göre ya simetriktir ya da asimetriktir. Toplamalı katsayılar, sayısal filtrelerde dürtü cevap katsayılarını tanımlar. Dürtü yanıt katsayılarından da frekans cevabı elde edilir.



$$y_n = \sum_{i=0}^{i=m} a_i . x_{n-i}$$

Şekil 2.12. FIR Filtre Mimarisi

FIR filtreler simetrik ve asimetrik olarak katsayılarının birleşmesiyle gerçekleşir. FIR filtreler farklı olarak dört çeşit tasarlanabilir. Her bir tip filtre tek karakteristik çözüme sahiptir.

1.Tip FIR filtreler simetrik katsayılara ve uzunluk olarak tek tamsayı değeri alırlar. Ayrıca kesim frekansları Ω=0 ve Ω=pi olan her ikisinde de çift uzunlukta simetrik olarak frekans cevabına sahiptirler. Bu çift simetride iki kritik frekans değerinde frekans cevabı alır. Bu çeşitle alçak geçiren, yüksek geçiren, band geçiren ve band durduran filtre tasarlanabilir.
2.Tip FIR filtreler simetrik katsayılara ve uzunluk olarak çift tamsayı değeri alanlardır. Çift olduğunda Ω=0, tek olduğunda Ω=pi olan frekans cevabına sahiptirler. Bu filtreler yüksek geçiren ve band durduran filtrelerde kullanılır.

3. ve 4. tip FIR filtreler asimetrik katsayılara sahiptirler. 3. tip FIR filtre tek uzunlukta ve frekans cevabı Ω=0 ve Ω=pi tek simetriye sahiptir. 4.tip FIR filtre ise çift uzunluktadır. Frekans cevabı Ω=0 dayken tek simetri ve Ω=pi dayken çift simetriye sahiptir[25].

2.3.1. Fourier serisi yöntemi

Filtre tasarımındaki komut dizisi, tasarımın frekans karakteristikleriyle başlar. Kritik kenar band frekansları ve her bandtaki kazançlar kesin tanımlamaları gösterir. Örnekleme frekans periyoduyla frekans periyodik olarak sayısal filtrelerin frekansı bulunur.

Frekans cevabı $H(e^{j.\Omega})$ olan sayısal bir filtre, Ω sayısal frekansının periyodik bir fonksiyonu olup 2π'lik periyoda sahiptir. Fourier serisi olarak periyodik olan $H(e^{j.\Omega})$ frekans cevabı yazılırsa,

$$H(e^{j.\Omega}) = \sum_{n=-\infty}^{\infty} h(n).e^{-jn\Omega} \qquad (2.5)$$

olur. Bu frekans cevabına denk gelen ideal dürtü cevabı ise

$$h(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H(e^{j\Omega}).e^{jn\Omega}.d\Omega \qquad (2.\ 6)$$

ifadesidir. Frekans cevabına karşılık gelen transfer fonksiyonu hesaplayabilmek için denklem 2.7'de $e^{j\Omega} = z$ dönüşümü yapılarak gerçekleştirilebilir.

$$H(z) = \sum_{n=-\infty}^{\infty} h(n).z^{-n} \qquad (2.\ 7)$$

Elde edilen sonsuz dürtü cevabı yani $-\infty$ dan $+\infty$ 'a kadar devam eder. Aralığın belli olmamasından dolayı filtre fiziksel olarak gerçekleştirilemez. Bir filtrenin fiziksel olarak gerçekleştirilebilmesi için dürtü cevabının sınırlı sayıda gerçekleştirilmesi gerekir. Sonsuz uzunluktaki dürtü cevabının sonlu uzunluktaki cevaba dönüştürülmesi için gecikmeyle çarpılması gerekir. Sonlu uzunlukta bir filtrenin dürtü cevabı için ise denklem 2.8 olarak gerçekleştirilir.

$$h(n) = \begin{cases} h(n) & |n| \leq N' = \dfrac{N-1}{2} \\ 0, & |n| > N' = \dfrac{N-1}{2} \end{cases} \qquad (2.\ 8)$$

Denklem (2.9) olarak kabul edildiğinde transfer fonksiyonu,

$$H(z) = h(0) + \sum_{n=1}^{N'} [h(-n).z^{n} + h(n).z^{-n}] \qquad (2.\ 9)$$

şeklinde olur. Transfer fonksiyonda nedenselliğin sağlanabilmesi için $z^{-N'}$ ile çarpışması yoluyla gerçekleştirilebilir.(2.10)

$$H'(z) = z^{-N'} H(z) \qquad (2.\ 10)$$

Bu gecikme ile elde edilen $H'(z)$ ile sonsuz uzunluktaki $H(z)$'nin genlik cevapları aynıdır. Fakat grup gecikmesinin faz cevabı üzerinde $N'T$ periyot olarak kaydırılması gerekir. [25]

2.3.2. İdeal FIR filtre cevap katsayıları

2.3.2.1. Normalleştirilmemiş alçak geçiren yanıt katsayıları

Zaman bölgesinde, alçak geçiren frekans bölgesinin cevabı, sinc(x) fonksiyonuna dönüşmektedir. Alçak geçiren yanıtların normalleştirilmemişi basittir. Kesim frekansı(ω_c'deki kesim) normalleştirilmiş yanıt 1 Hz'lik(2π radyan/sn) bir örnekleme hızına göre değişmektedir.

Alçak geçiren filtrenin katsayıların hesaplanmasında integral form için denklem 2.11 kullanılır. Filtrenin uzunluğunu belirleyen n değeri 1 den başlayarak 2M kadar devam eder.

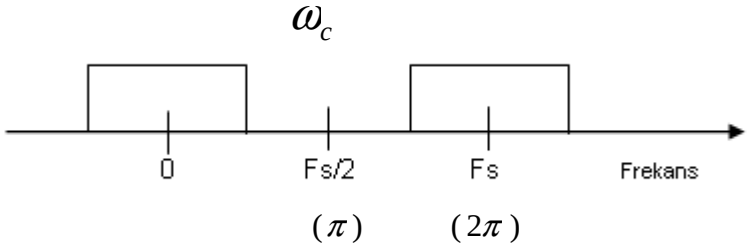
$$h_{ag}(n) = \frac{1}{2\pi} \int_{-\Omega_c}^{+\Omega_c} e^{j(n)\Omega} . d\Omega \qquad (2.\ 11)$$

Alçak geçiren filtrenin katsayıların hesaplanmasında sinc(x) form kullanılarak bulunması için denklem 2.12 kullanılır

ω_c (alçak geçim frekansı) değeri:

$$\omega_c = \frac{f_c}{f_s} \qquad (2.\ 12)$$

denklemini (2.11) sağlaması gerekir. f_c değeri ise kesim frekansı, f_s ise örnekleme hızını göstermektedir.



Şekil 2.13. Örneklenmiş Alçak Geçiren Frekans Cevabı

Şekil 2.13'de alçak geçiren filtreler için örnekleme frekansı ile filtrenin kesim frekansı arasındaki ilişkiyi gösterir.

a)Orta katsayısı değeri;

$h[0] = \frac{\omega_c}{\pi}$ denkleminden gerçekleştirilir. (2. 13)

b)Diğer katsayıların değeri;

$h[n] = \frac{\sin(\omega_c n)}{\pi n}$ denkleminden gerçekleştirilir. (2. 13)

Örneğin: İstenen filtrenin değerleri F_c=3.2 kHz ve F_s=15 kHz olsun. Bu değerlerden alçak geçiren filtrenin yanıt katsayılarını bulunuz.

İlk adım olarak kesim frekansı elde edilir. Daha sonraki adım da orta katsayı ve diğer katsayılar elde edilmiştir.

$\omega_c = \frac{F_c}{F_s}$ ω_c=3.2/15=0.2133

a) h[0]=$\frac{\omega_c}{\pi} = \frac{0.2133}{3.14}$=0.06761

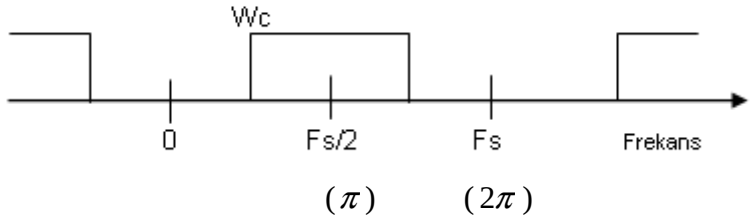
b) $h[1] = \frac{\sin(\omega_c n)}{\pi n} = \frac{\sin(0.2133*1)}{1*3.14}$=0.06738

c) $h[2] = \frac{\sin(\omega_c n)}{\pi n} = \frac{\sin(0.2133*2)}{2*3.14} = 0.06585$ katsayı değerleri elde edilebilir.

Katsayıların değerleri verilen değerlere göre değişebilir.

2.3.2.2. Normalleştirilmemiş yüksek geçiren yanıt katsayıları

Zaman bölgesinde, yüksek geçiren frekans bölgesinin cevabı, negatif sinc(x) fonksiyonuna dönüşmektedir. Yüksek geçiren filtrenin normalleştirilmemiş alçak geçiren filtreninkine benzerdir. Kesim frekansı(ω_c'deki kesim) normalleştirilmiş yanıt 1 Hz'lik(2π radyan/sn) bir örnekleme hızına göre değişmektedir.



Şekil 2.14. Örneklenmiş Yüksek Geçiren Frekans Cevabı

Yüksek geçiren filtrenin katsayıların hesaplanmasında integral form için denklem 2.13 kullanılır Filtrenin uzunluğunu belirleyen n değeri 1 den başlayarak 2M kadar devam eder.

$$h_{yg}(n) = \frac{1}{2\pi}\left[\int_{-\pi}^{-\Omega_c} e^{j(n)\Omega}.d\Omega + \int_{\Omega_c}^{\pi} e^{j(n)\Omega}.d\Omega\right] \quad (2.14)$$

Yüksek geçiren filtrenin katsayıların hesaplanmasında sinc(x) form kullanılarak bulunması için denklem 2.15 kullanılır

w_c (yüksek kesim frekansı) değeri:

$$\omega_c = \frac{f_c}{f_s} \quad (2.15)$$

denklemini sağlaması gerekir.

a)Orta katsayısı değeri;

$h[0] = 1 - \frac{\omega_c}{\pi}$ denkleminden gerçekleştirilir. (2. 15)

b)Diğer katsayıların değeri;

$h[n] = -\frac{\sin(\omega_c n)}{\pi n}$ denkleminden gerçekleştirilir. (2. 15)

Örneğin: İstenen filtrenin değerleri f_c=5 kHz ve f_s=10 kHz olsun. Bu değerlerden elde edilen yüksek geçiren filtrenin yanıt katsayılarını bulunuz.

İlk adım olarak kesim frekansı elde edilir. Daha sonraki adım belirlenen uzunlukta katsayıların elde edilmesidir.

$\omega_c = \frac{f_c}{f_s}$ ω_c=5/10=0.5

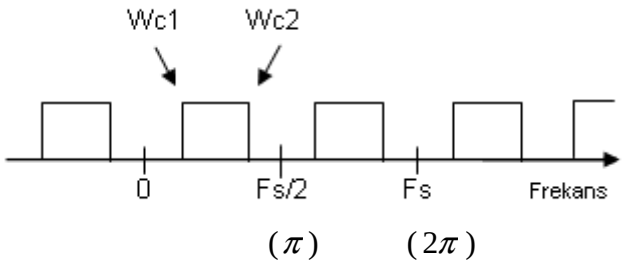
a) $h[0] = 1 - \frac{\omega_c}{\pi} = 1 - \frac{0.5}{3{,}14}$ =0,84085

b) $h[1] = -\frac{\sin(\omega_c n)}{\pi n} = \frac{\sin(0.5*1)}{\pi*1}$ =0.15260*-1=-0.1526

c) $h[2] = -\frac{\sin(\omega_c n)}{\pi n} = \frac{\sin(0.5*2)}{\pi*2}$ =0.133924*-1=-0.133924 katsayı değerleri elde edilebilir. Burada katsayıların değerleri verilen değerlere göre değişebilir.

2.3.2.3. Normalleştirilmemiş bant geçiren yanıt katsayıları

Zaman bölgesinde değiştirilmiş, yüksek geçiren frekans bölgesinin cevabı, sinc(x) fonksiyonuna dönüşmektedir. Bant geçiren cevabın normalleştirilmemiş için kesim frekansının alt ve üst bant sınırları belirlenmelidir. Normalleştirilmişte yanıtta örnekleme hızı 1 Hz(2π radyan/sn) dir. Kesim frekansları ωc_1 ve ωc_2'ne bağlıdır. Kesim frekanslarından ωc_1 alt sınır kesim noktasını, ωc_2 üst sınır kesim noktasını gösterir.



Şekil 2.15. Örneklenmiş Bant Geçiren Frekans Cevabı

Band geçiren filtrenin katsayıların hesaplanmasında integral form için denklem 2.16 kullanılır. Filtrenin uzunluğunu belirleyen n değeri 1 den başlayarak 2M kadar devam eder.

$$h_{bg}(n) = \frac{1}{2\pi}\left[\int_{-\Omega_{c2}}^{-\Omega_{c1}} e^{j(n)\Omega}.d\Omega + \int_{+\Omega_{c1}}^{+\Omega_{c2}} e^{j(n)\Omega}.d\Omega\right] \quad (2.14)$$

Band geçiren filtrenin katsayıların hesaplanmasında sinc(x) form kullanılarak bulunması için denklem 2.17 kullanılır

ω_{c1} ve ω_{c2} (alt ve üst kesim frekans) değerleri:

$$\omega_{c1} = \frac{F_{c1}}{F_s} \qquad \omega_{c2} = \frac{F_{c2}}{F_s} \quad (2.15)$$

denklemlerinden bulunur.

a)Orta katsayısı değeri;

$h[0] = \frac{\omega_{c2} - \omega_{c1}}{\pi}$ denkleminden gerçekleştirilir. (2. 17)

b)Diğer katsayıların değeri;

$h[n] = \frac{\sin(\omega_{c2}n) - \sin(\omega_{c1}n)}{\pi n}$ denkleminden gerçekleştirilir. (2. 17)

Örneğin**:** İstenen filtrenin değerleri f_{c1}=3 kHz ve f_{c2}=6 kHz ve f_s=18 kHz olsun. Bu değerlerden elde edilen bant geçiren filtrenin yanıt katsayılarını bulunuz.

İşlem basamaklarının en başında kesim frekansları elde edilir ve sonra da elde edilen kesim frekansları kullanılarak istenen band geçiren filtrenin orta ve diğer katsayılar elde edilir.

$$\omega_{c1} = \frac{F_{c1}}{F_s} = \frac{3}{18} = 0.167 \qquad \omega_{c2} = \frac{F_{c2}}{F_s} = \frac{6}{18} = 0{,}333$$

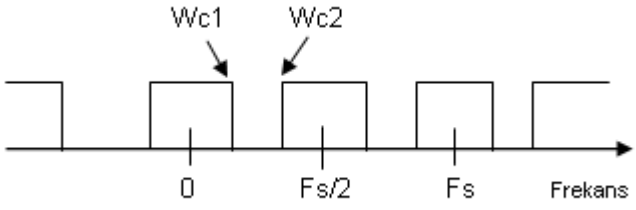
a) $h[0] = \dfrac{\omega_{c2} - \omega_{c1}}{\pi} = \dfrac{0.333 - 0.167}{\pi} = 0.0530$

b) $h[1] = \dfrac{\sin(\omega_{c2} n) - \sin(\omega_{c1} n)}{\pi n} = \dfrac{\sin(0.333 * 1) - \sin(0.167 * 1)}{\pi * 1} = 0.0513$

c) $h[2] = \dfrac{\sin(\omega_{c2} n) - \sin(\omega_{c1} n)}{\pi n} = \dfrac{\sin(0.333 * 2) - \sin(0.167 * 2)}{\pi * 2} = 0.0463$ katsayı değerleri elde edilebilir. Burada katsayıların değerleri verilen değerlere göre değişebilir.

2.3.2.4. Normalleştirilmemiş bant durdurucu yanıt katsayıları

Zaman bölgesinde değiştirilmiş bir sinc(x) fonksiyonu bant durdurucu frekans bölgesinin cevabıdır. Bant durdurucu katsayılar veren denklem ile bant geçiren katsayılarını veren denklem arasındaki ilişki alçak geçiren katsayı denklemlerinin yüksek geçiren katsayıları vermek üzere değiştirilmesine benzer bir ilişki vardır. Bant durdurucu cevabın normalleştirilmemiş için kesim frekanslarının alt ve üst sınırlarının belirlenmesi gerekir. Normalleştirilmişte yanıtta örnekleme hızı 1 Hz(2π radyan/sn) dir. Kesim frekansları ω_{c1} ve ω_{c2}'ne bağlıdır. Kesim frekanslarından ω_{c1} alt sınır kesim noktasını, ω_{c2} üst sınır kesim noktasını gösterir.



(π) (2π)

Şekil 2.16. Örneklenmiş Bant Durduran Frekans Cevabı

Band durduran filtrenin katsayıların hesaplanmasında integral form için denklem 2.18 kullanılır. Filtrenin uzunluğunu belirleyen n değeri 1 den başlayarak 2M kadar devam eder.

$$h_{bd}(n) = \frac{1}{2\pi}\left[\int_{-\pi}^{-\Omega_{c2}} e^{j(n)\Omega}.d\Omega + \int_{-\Omega_{c1}}^{+\Omega_{c1}} e^{j(n)\Omega}.d\Omega + \int_{+\Omega_{c2}}^{\pi} e^{j(n)\Omega}.d\Omega\right] \qquad (2.16)$$

Band durduran filtrenin katsayıların hesaplanmasında sinc(x) form kullanılarak bulunması için denklem 2.19 kullanılır

ω_{c1} ve ω_{c2} (kesim frekansları) değerleri:

$$\omega_{c1} = \frac{f_{c1}}{f_s} \qquad \omega_{c2} = \frac{f_{c2}}{f_s} \qquad (2.17)$$

denklemlerinden bulunur.

a)Orta katsayısı değeri;

$h[0] = 1 - \frac{\omega_{c2} - \omega_{c1}}{\pi}$ denkleminden gerçekleştirilir. (2. 18)

b)Diğer katsayıların değeri;

$h[n] = \frac{\sin(\omega_{c1}n) - \sin(\omega_{c2}n)}{\pi.n}$ denkleminden gerçekleştirilir. (2. 19)

Örneğin**:** İstenen filtrenin değerleri f_{c1}=3 kHz ve f_{c2}=6 kHz ve f_s=18 kHz olsun. Bu değerlerden elde edilen bant durduran filtrenin yanıt katsayılarını bulunuz.

İşlem basamaklarının en başında kesim frekansları elde edilir ve sonra da elde edilen kesim frekansları kullanılarak istenen band geçiren filtrenin orta ve diğer katsayılar elde edilir.

$$\omega_{c1} = \frac{F_{c1}}{F_s} = \frac{3}{18} = 0.167 \qquad \omega_{c2} = \frac{F_{c2}}{F_s} = \frac{6}{18} = 0{,}333$$

a) $h[0] = 1 - \frac{\omega_{c2} - \omega_{c1}}{\pi} = 1 - \frac{0.333 - 0.167}{\pi} = 0.9472$

b) $h[1] = \frac{\sin(\omega_{c1} n) - \sin(\omega_{c2} n)}{\pi n} = \frac{\sin(0.167 * 1) - \sin(0.333 * 1)}{\pi * 1} = -0.0511$

c) $h[2] = \frac{\sin(\omega_{c1} n) - \sin(\omega_{c2} n)}{\pi n} = \frac{\sin(0.167 * 2) - \sin(0.333 * 2)}{\pi * 2} = -0.0462$

katsayı değerleri elde edilebilir. Burada katsayıların değerleri verilen değerlere göre değişebilir.

2.4. Sonlu Dürtü Yanıt Filtrelerin Avantajları

1- FIR filtre, sonlu dürtü cevabına sahip olduğu için form üzerine simetrik çizilir. Bu da ideal lineer faz karakteristiğini ortaya koyar. Bunun sonucu olarak, filtreden geçen bütün frekans bileşenlerine eşit derecede saf zaman gecikmesi uygulandığı sonucuna varılır.

2- FIR filtreler, kutup olmadığı için doğal olarak kararlıdır. Filtrenin transfer fonksiyonunu z düzleminde bulunan yalnızca sıfır terimleri karakterize eder.

3- FIR filtrelerin tasarım metotları genelde lineer, basit ve güvenilirdir. Çoğu sayısal işleme sayısal sinyal işleme mikroişlemcilerinde FIR hesabı, bir çevrim sayesinde yapılabilir.

4- FIR filtreler, herhangi bir şekilde geri besleme elemanına sahip olmadıkları için, herhangi sınırlı girişe karşılık, sınırlı çıkış üretirler.

2.5. Pencereleme Tekniği

İdeal filtrelerin gerçekleştirilmesinde sonsuz sayıda katsayı kullanılmamıştır. Tasarımda belirtilen sınırları baz tutarak katsayı sayısı azaltılmıştır. Önceki konuda

katsayıların hesaplanmasında dizinin indisleri $\pm M$ olarak sınırlandırılmıştır. Pencerelemede sınırlar içerisindeki bütün katsayılar elde tutulur ve sınırlar dışındaki yok sayılır.

Alçak geçiren cevabı ve diğer yanıtlar kullandığı denklemde sinc fonksiyonun genişliği artı(+) ve eksi (-) sonsuza uzanmaktadır. Bu genişlik belli bir bölgeyle sınırlandırıldığında yani zamana sınırlama koyulduğunda, bant zayıflaması sınırlandırılmakta ve frekans cevabının duran ve geçen bantlarında dalgacıklara neden olmaktadır. Bu genişliğin kesilmesi pencere fonksiyonlarıyla dereceli olarak uygulanabilir. Derecelendirme yoluyla dalgacık etkileri azaltmakta ve duran bant zayıflamasını iyileştirilebilir. Pencereleme işlemi, sinc(x) fonksiyonuyla pencere katsayılarıyla çarpılmasıyla gerçekleştirilir.

Pencereler, sinc(x) fonksiyonun belirli sayıda ara uç bağlantılarıyla sınırlandırılarak oluşturulur. Eğer tek sayıda N ara uç bağlantısının orta noktası sıfır zaman noktasıdır. Periyotlar sıfır noktasına göre -(N-1)/2 ve +(N+1)/2 örnek periyotlar arasında gerçekleştirilir. Bunun nedeni olarak da sinc(x) fonksiyonun sahip olduğu ideal frekans bölgesi artı ve eksi sonsuza uzanan bir yanıta sahiptir. Örneğin 23 ara bağlantılı bir filtrede, örneklemeyi 1ms için alırsak sıfır noktasınında 11 ms'de olmalıdır.

Eğer bölgede negatif zaman olması istenmiyorsa filtrenin gerçekleştirilmesinde gecikme gerçekleştirilir. Pencereler genel olarak simetriktir. Bu yüzden sıfır zaman noktasından pozitif zaman bölgesindeki katsayılar bulunur. Bu uç bağlantılar da $\omega(0)$ ile $\omega([N-1]/2)$ arasındadır.

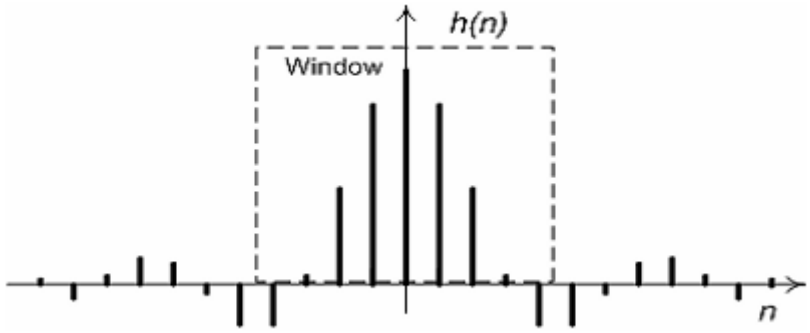
Tek sayıda ara uç bağlantıda, sıfır zamanın her iki yanındaki katsayılar birbirine simetriktir. Bu simetriyi de $\omega([N-1]/2-1) = \omega([N-1]/2+1)$ formülüyle eşitlik sağlanabilir.

Çift sayıda ara uç bağlantıda, sıfır zaman noktasındaki katsayı iki sayının arasında kaldığı için bu orta nokta sıfır zaman noktası olarak adlandırılır. Simetrik ara uç bağlantılar ise $\omega([N–2]/2) = \omega([N–2]/2-1)$ denklemiyle sağlanır.

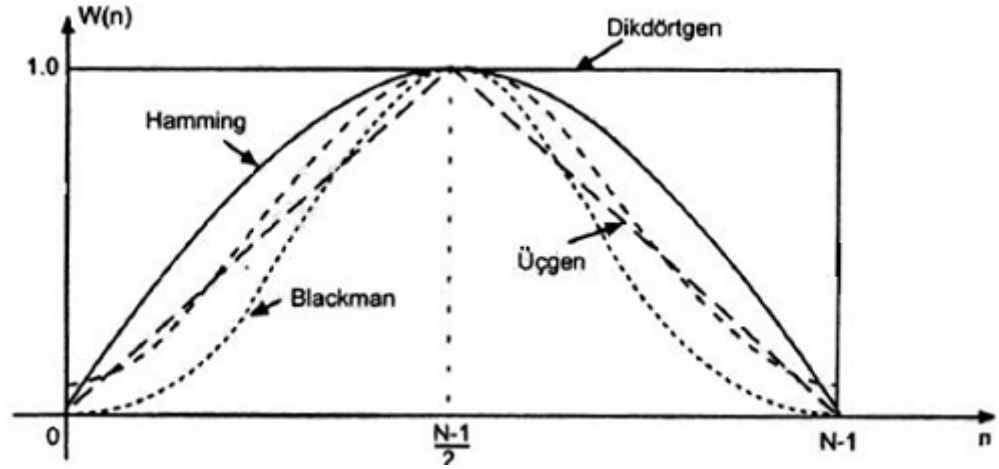
Pencereleme tekniğiyle FIR filtreleri tasarlamayı özetlersek;

— İdeal filtrelerde ters fourier dönüşümü sonsuz uzunlukta olan bir sinc fonksiyonudur. Filtrenin sonsuz uzunlukta olması yerine sonlu sayıda örnekle yaklaşılır. Pencereleme yönteminde tasarımı matematiksel olarak ifade ederek filtrenin h(n) dürtü(dürtü) cevabı ve w(n) pencereleme fonksiyonu ile çarpılır(Denklem 22). Şekil 2.17 de görüldüğü nedensel olmayan katsayıları pencereleme yöntemiyle nedensel hale getirilmiştir. Filtre tasarımında kullanılan pencereleme fonksiyonları şekil 2.18 de gösterilmiştir.

$$h(n) = h_{ideal}(n).\omega(n), \;\; n = 0,\pm,1,...,\pm M \qquad (2.\ 20)$$



Şekil 2. 17. Filtre dürtü cevabının pencereleme metodu ile sınırlanması



Şekil 2. 18. Filtre tasarımında kullanılan pencere fonksiyonları

2.5.1. Pencere tipleri

1. Dikdörtgensel pencere

Dikdörtgensel pencereleme de indis aralığındaki bütün katsayıların değerleri 1'dir sınır dışındakiler ise 0'dır.

ω(n)=1.0, n=-(N-1)/2,…,-1,0,1,2,…,(N-1)/2.

Dikdörtgensel pencerede yan lob bant zayıflaması 13.2 dB dir. Frekansların ikiye katlanması durumunda ise artış miktarı 6 dB'dir.

2. Üçgensel pencere

Üçgensel pencerede sıfır noktasından sağ ve sola doğru azalan katsayılara sahiptir. Dikdörtgensel pencerede yan lob bant zayıflaması 27 dB dir. Frekansların ikiye katlanması durumunda ise artış miktarı 12 dB'dir. Diğer noktaların katsayı değerlerini hesaplayabilmek için, katsayı değerinin sıfır olarak ayarlanarak gerçekleştirilebilir. Dizi indisleri n=-(N-1)/2, -(N-3)/2, …-1,0,1,… (N-3)/2, (N-1)/2. belirtilen şekilde ayarlandığında simetrik olarak işlem gerçekleştirilebilir. Üçgensel pencerelemenin katsayıları denklem 2.23 kullanılarak hesaplanır.

$$\omega(n) = 1.0 - |n| / (\frac{N-1}{2}) \qquad (2.\ 21)$$

3. Von Hann (yükseltilmiş kosinüs) pencere

Von Hann penceresi, katsayıları kosinüs fonksiyonunun karesinden faydalanarak hesapladığından dolayı bir diğer adı da yükseltilmiş kosinüsdür.

Basitleştirilmiş kosinüs ifade $h(n) = \cos^2\left[\frac{n\pi}{N}\right]$ bulunur. Katsayıları hesaplarken kullanılan diğer bir ifade ise $h(n) = 0.5 + 0.5\cos(\frac{2\pi n}{N})$ dir. Aralık adımları;

n= -(N–1)/2, ...-1,0,1, …(N-3)/2,(N-1)/2 dir.

Eğer pencerede zamanın sadece pozitif kısmı yani n=0 ve n=N–1 aralığı şeklinde zaman kayması gerçekleştirilirse denklemin pozitif işareti değiştirilerek negatif olmaktadır. $h(n) = 0.5 - 0.5\cos(\frac{2\pi.(n+1)}{N+1})$ buradaki aralık ise n= 0,1, …(N–3)/2,(N–1)/2' dir. Paydaki n değerine 1 eklenmesinin nedeni ise uç değerlerde sıfır değerinin olmamasından kaynaklanmaktadır. Paydaki N+1 nedeni ise n=(N–1)/2 sıfır zaman değerini yok etmek için paydaya 1 eklenerek h(n)=1 sağlanmaktadır.
Von Hann penceresinde yan lob için bant zayıflaması 35 db dir ve yüksek frekansların ikiye katlanmasıyla her artışta değer 18 db arttırılır.

4. Hamming pencere

Von Hann penceresinde bant zayıflamasında iyileştirilebilmesi için yan lob da 43 db'ye ulaştırılır. Frekansların ikiye katlanması durumunda ise artış miktarı 6 dB'dir.

Katsayı değerlerini hesaplanması için aralık olarak n= -(N–1)/2, ...-1,0,1, …(N-1)/2 kullanıldığında $h(n) = 0.54 + 0.46\cos\left[\frac{2\pi n}{N}\right]$ dir. Eğer pencerede aralığın n= 0,1, …(N-1)/2 olması istendiğinde yani pozitif zaman kayması gerçekleştirildiğinde denklemin ortasındaki işaret değişmektedir. Denklem ise $h(n) = 0.54 - 0.46\cos\left[\frac{2\pi n}{N}\right]$ dir.

5. Blackman pencere

Kosinüs serisine ek olarak denklem eklediğinde blackman penceresini gösterir. Blackman penceresinde aralık n= -(N–1)/2, ...-1,0,1, …(N-1)/2 olduğu zaman denklem $h(n) = 0.42 + 0.5\cos\left[\frac{2\pi.n}{N}\right] + 0.08\cos\left[\frac{2\pi.n}{N}\right]$ dir.

Eğer zaman kayması gerçekleştirildiğinde yani n=0 ile n=N–1 arasındaki aralıkta denklemin ortasındaki işaret değişmektedir. Pencere sınırlarında sıfır değerli katsayıları engellemek için pay ve payda değerleri 1 birim arttırılır. Aralık olarak

n=0,1, …(N–1)/2 ise $h(n) = 0.42 - 0.5\cos\left[\frac{2\pi.(n+1)}{N+1}\right] + 0.08\cos\left[\frac{4\pi.(n+1)}{N+1}\right]$

denklemi kullanılır.

6. Blackman-Harris pencere

Harris penceresi blackman penceresinde var olan duran bant zayıflamasını iyileştirebilmek için üç terimli kosinüs serisini düzenler. Üç terimli seri katsayıları tarafından üretilen ilk yan lob bant zayıflaması 61 dB'dir. Frekans yükseldikçe zayıflama da artmaktadır. Blackman harris dört terimli katsayı özelliği de vardır ve dört terimli seri üç terimli seriye göre daha iyi duran bant zayıflaması gerçekleştirir. Dört terimli seri ilk yan lob bant zayıflaması 74 dB'dir.

a) Üç terimli Blackman-Harris katsayıları:

Zaman aralığı n= -(N-1)/2, ...-1,0,1, …, (N-1)/2 ise;

$h(n) = 0.44959 + 0.49364\cos[\frac{2\pi.n}{N}] + 0.05677\cos\left[\frac{4\pi.n}{N}\right]$ denklemi kullanılır.

Zaman kaydırması gerçekleştirildiğinde aralık n=0,1, …, (N-1)/2 olduğunda denklem;

$h(n) = 0.44959 - 0.49364\cos[\frac{2\pi.n}{N}] + 0.05677\cos\left[\frac{4\pi.n}{N}\right]$ dir.

b) Dört terimli Blackman-Harris katsayıları:

Zaman aralığı n= -(N-1)/2, ...-1,0,1, …(N-1)/2 ise

$h(n) = 0.40217 + 0.49703\cos\left[\frac{2\pi.n}{N}\right] + 0.09892\cos\left[\frac{4\pi.n}{N}\right] + 0.00183\cos\left[\frac{6\pi.n}{N}\right]$

denklemi kullanılır.

Zaman kaydırması gerçekleştirildiğinde aralık n=0,1, …, (N-1)/2 olduğunda denklem;

$$h(n) = 0.40217 - 0.49703\cos\left[\frac{2\pi.n}{N}\right] + 0.09892\cos\left[\frac{4\pi.n}{N}\right] - 0.00183\cos\left[\frac{6\pi.n}{N}\right]$$

şeklindedir.

7. Harris-Nutall pencere

Harris-Nutall katsayıları hesaplarken duran bant zayıflamasını Blackman-Harris göre daha iyileştirir.

a) Üç terimli Harris-Nutall katsayıları:

Zaman aralığı n= -(N-1)/2, ...-1,0,1, …(N-1)/2 ise

$h(n) = 0.42323 + 0.49755\cos\left[\frac{2\pi.n}{N}\right] + 0.07922\cos\left[\frac{4\pi.n}{N}\right]$ denklemi kullanılır.

Zaman kaydırması gerçekleştirildiğinde aralık n=0,1, …, (N-1)/2 olduğunda denklem;

$h(n) = 0.42323 - 0.49755\cos\left[\frac{2\pi.n}{N}\right] + 0.07922\cos\left[\frac{4\pi.n}{N}\right]$ şeklindedir.

b) Dört terimli Harris-Nutall katsayıları:

Zaman aralığı n= -(N-1)/2, ...-1,0,1, …(N-1)/2 ise

$$h(n) = 0.35875 + 0.48829\cos\left[\frac{2\pi.n}{N}\right] + 0.14128\cos\left[\frac{4\pi.n}{N}\right] + 0.01168\cos\left[\frac{6\pi.n}{N}\right]$$

denklemi kullanılır.

Zaman kaydırması gerçekleştirildiğinde aralık n=0,1, …, (N-1)/2 olduğunda denklem;

$$h(n) = 0.35875 - 0.48829\cos\left[\frac{2\pi.n}{N}\right] + 0.14128\cos\left[\frac{4\pi.n}{N}\right] - 0.01168\cos\left[\frac{6\pi.n}{N}\right]$$

şeklindedir.

8. Kaiser-Bessel pencere

Kaiser penceresi sabit bir pencere değildir ve farklı duran bant zayıflamalarına sahiptir. Bant zayıflamalarını değiştirecek α faktörüyle değiştirilen bir formül geliştirilmiştir. Bu α faktörü 0 ila 4 arasında olmalıdır. h(n) için denklem sabit değerleri $\alpha(0) = \frac{H_1(0)}{c}$ ve $\alpha(m) = \frac{2H_1(m)}{c}$ dir. m=1,2,3 için $H_1(m) = \frac{\sinh(\pi\sqrt{a^2 - m^2})}{\pi\sqrt{a^2 - m^2}}$ ve $c = H(0) + 2.H(1) + 2.H(2) + 2H(3)$ dir. Kaiser penceresinde ilk yan lob bant zayıflaması 70 db'dir.

Zaman aralığı n= -(N-1)/2, ...-1,0,1, …(N-1)/2 ise $h(n) = 0.40243 + 0.49804\cos\left[\frac{2\pi.n}{N}\right] + 0.09831\cos\left[\frac{4\pi.n}{N}\right] + 0.00122\cos\left[\frac{6\pi.n}{N}\right]$ dir.

Zaman kaydırması gerçekleştirildiğinde aralık n=0,1, …, (N-1)/2 olduğunda denklem;

$$h(n) = 0.40243 - 0.49804\cos\left[\frac{2\pi.n}{N}\right] + 0.09831\cos\left[\frac{4\pi.n}{N}\right] - 0.00122\cos\left[\frac{6\pi.n}{N}\right]$$

dir[24].

BÖLÜM 3. GENETİK ALGORİTMALAR

3.1. Giriş

Genetik Algoritmalar evrimsel hesaplamanın bir parçasıdır. Bu alan Yapay Zekâ'nın hızla gelişen bir dalıdır. Genetik algoritmalar Darwin' in evrim teorisinden etkilenerek geliştirilmiştir. Basitçe açıklayacak olursak problemler evrimsel bir süreç kullanılarak bu süreç sonunda en iyi sonucu veren çözüme erişmeye çalışmaktadır. Başka bir ifadeyle çözüm evrimleşmektedir.

Evrimsel hesaplama 1960'larda I.Rechenberg'in Evrim Stratejileri (*Evolutionsstrategie)* adlı çalışmasında tanıtılmıştır. Daha sonra fikri diğer araştırmacılar tarafından geliştirilmiştir.

Genetik algoritmalar John Holland tarafından icat edilmiş ve öğrencileri ve meslektaşları tarafından geliştirilmiştir (Holland, 1975). Mekanik öğrenme (machine learning) konusunda çalışan Holland, Darwin'in evrim kuramında etkilenerek canlılarda yaşanan genetik süreci bilgisayar ortamında gerçekleştirmeyi düşündü. Tek bir mekanik yapının öğrenme yeteneğini geliştirmek yerine böyle yapılarda oluşan bir topluluğun çoğalma, çiftleşme, mutasyon, vb. genetik süreçlerden geçerek başarılı (öğrenebilen) yeni bireyler oluşturabildiğini gördü. Araştırmalarını, arama ve optimumu bulma için, doğal seçme ve genetik evrimden yola çıkarak yapmıştır. İşlem boyunca, biyolojik sistemde bireyin bulunduğu çevreye uyum sağlayıp daha uygun hale gelmesi örnek alınarak, optimum bulma ve makine öğrenme problemlerinde, bilgisayar yazılımı modellenmiştir.

1992 yılında John Koza, genetik algoritmaları kullanarak programları evrimleştirerek belli işleri yapmakta kullanmıştır. Bu yönteme "genetik programlama" adını verdi.

LISP dilinde programlar Ayrıştırma Ağaçları" ("Parse Tree") seklinde ifade edildiği için LISP diliyle geliştirilmiştir. Ayrıştırma Ağaçları genetik algoritmaların çalıştığı temel nesnedir.

3.2. Genetik Algoritmanın Tanımı

Algoritma, araştırma uzayında mevcut olan çözümlerin oluşturduğu bir başlangıç yoğunluğunu (popülasyon) kullanır. Bu başlangıç yoğunluğu, her bir kuşakta (jenerasyon) tabii seçme ve tekrar üreme işlemleri vasıtası ile art arda geliştirilir. En son kuşağın en uygun yani en kaliteli bireyi, problem için optimal bir çözümdür. Bu çözüm, her zaman optimum olmayabilir, ancak kesinlikle optimuma en yakın olan çözümdür. Çok boyutlu bir araştırma uzayında, global optimum bir çözümü rahatlıkla bulabilmektedir.

Genetik algoritmanın paralel işlem yapabilen bilgisayarlarda kullanılmaya elverişli yapısı, zaman alıcı problemlerin kısa zamanda çözümü için çekici bir alternatif olmasını sağlamıştır.

Genetik algoritma, klasik optimizasyon yöntemleri ile çözümü mümkün olmayan veya çözüm süresi problemin büyüklüğü ile üstel olarak artan problemlerin çözümünde etkin olarak kullanılmaktadır.

Genetik algoritmanın temel avantajı, optimize etmeye çalıştığı problemin tabiatı ile ilgili herhangi bir bilgiye ihtiyaç duymamasıdır. GA'nın farklı problemlere uyarlanmasında en önemli adım, probleme özgü genetik kodlama ve uygunluk fonksiyonunun belirlenmesidir.

GA'ların probabilistik karakterleri ve çoklu mümkün çözümleri araştırma gibi önemli özelliklere sahip olmaları ve amaç fonksiyonunun gradyanının bilinmesine ihtiyaç duymamaları en önemli üstünlükleridir. Diğer önemli özelliği de diğer algoritmalardan ayıran en önemli özelliklerinden biri de seçmedir.

Genetik Algoritmaları (GA) diğer metotlardan ayıran noktalar şu şekilde sıralanabilir:

- GA, sadece bir arama noktası değil, bir grup arama noktası (adaylar) üzerinde çalışır. Yani arama uzayında, yerel değil global arama yaparak sonuca ulaşmaya çalışır. Bir tek yerden değil bir grup çözüm içinden arama yapar.
- GA, arama uzayında bireylerin uygunluk değerini bulmak için sadece "amaç - uygunluk fonksiyonu" (objective-fitness function) ister. Böylelikle sonuca ulaşmak için türev ve diferansiyel işlemler gibi başka bilgi ve kabul kullanmaya gerek duymaz.
- Bireyleri seçme ve birleştirme aşamalarında deterministik kurallar değil " olasılık kuralları" kullanır.
- Diğer metotlarda olduğu gibi doğrudan parametreler üzerinde çalışmaz. Genetik Algoritmalar, optimize edilecek parametreleri kodlar ve parametreler üzerinde değil, bu kodlar üzerinde işlem yapar. Parametrelerin kodlarıyla uğraşır. Bu kodlamanın amacı, orijinal optimizasyon problemini kombinezonsal bir probleme çevirmektir.
- Genetik algoritma ne yaptığı konusunda bilgi içermez, nasıl yaptığını bilir. Bu nedenle kör bir arama metodudur.
- Olasılık kurallarına göre çalışırlar. Programın ne kadar iyi çalıştığı önceden kesin olarak belirlenemez. Ama olasılıkla hesaplanabilir.
- GA, kombinezonsal bir atama mekanizmasıdır.

Genetik algoritmalar, yeni bir nesil oluşturabilmek için 3 aşamadan geçer;

Eski nesildeki her bir bireyin uygunluk değerini hesaplama.

Bireyleri, uygunluk değerini göz önüne alarak (uygunluk fonksiyonu) kullanılarak seçme.

Seçilen bireyleri, çaprazlama (crossover), mutasyon (mutation) gibi genetik operatörler kullanarak uyuşturma.

Genetik Algoritmalar; başlangıçta bilinmeyen bir arama uzayından topladığı bilgileri yığıp, daha sonraki aramaları alt arama uzaylarına yönlendirmek için kullanılır.[29]

3.3. GA Terimleri

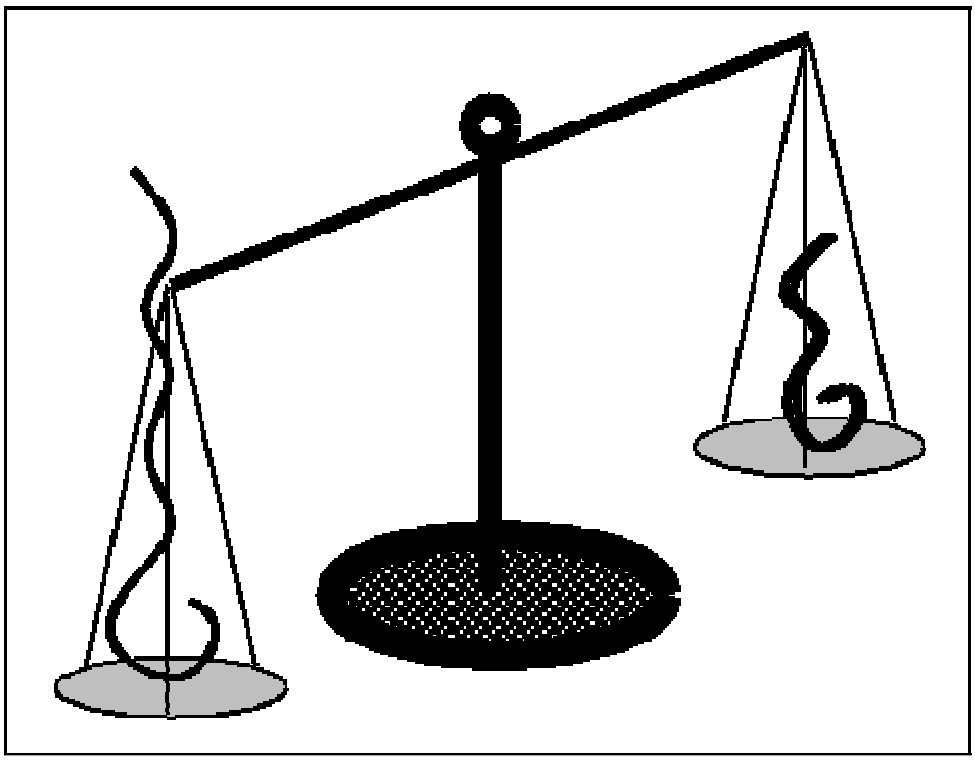
3.3.1. Kromozom ve gen

Bütün yaşayan canlılar hücrelerden oluşmuştur. Her hücrede bir grup kromozom vardır. Kromozomlar DNA dizileridir ve bütün organizma için bir model görevi görür. Kromozomlar genlerden oluşmuştur. Bir kromozomun elemanlarından her birisi çözümün bir özelliği göstermektedir. Bunlara da "**Gen**" adı verilir. Genlerde DNA bloklarından oluşmuştur. Her gen belli bir proteini kodlar. Temel olarak her gen bir özelliği kodlar. Örneğin; göz rengi vb. Bir özellik için olası ayarlar(kahverengi ve mavi) **alel** (aynı özellikleri taşıyan gen) olarak adlandırılır. Her gen kromozomda özel bir yeri vardır. Bu yerleşimin adı lokustur(özel gen bölgesi).

Genetik maddelerin tümü(tüm kromozomlar) **genom** olarak adlandırılır. Genomdaki genlerin belli bir grubu **genotip** olarak adlandırılır. Doğumdan sonraki gelişmeyle genotip organizmanın zihinsel ve fiziksel özellikleri temsil eden **fenotip** gelişimine zemin hazırlar. Örneğin; göz rengi, zekâ vb.

3.3.2. Tekrar üretim(Çoğalma)

Tekrar üretim sırasında, yeniden birleşme (veya çaprazlama) ilk önce ortaya çıkar. Ebeveynlerden gelen genler ile yeni bir kromozomlar oluşturulur. Bu yeni yaratılmış nesil daha sonra mutasyona uğrayabilir. **Mutasyon DNA** elemanlarının değişmesidir. Bu değişimler genellikle atalardan gen kopyalanması sırasındaki hatalardan kaynaklanır. Bir organizmanın uygunluğu ("fitness") organizmanın yaşamındaki başarısıyla (hayatta kalma) ölçülür.

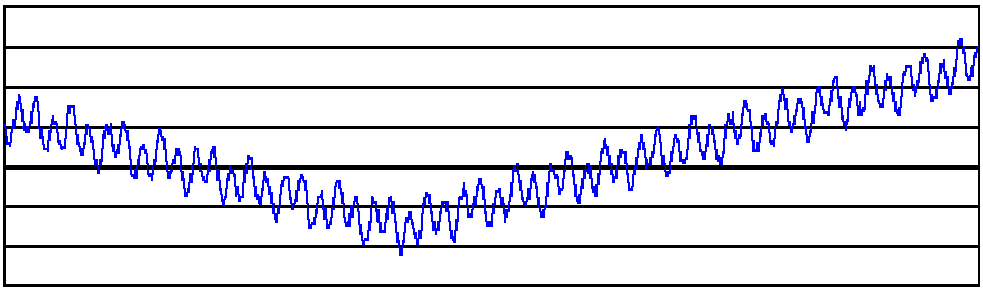


Şekil 3.1. Değerlendirme ile hangi bireyin uygunluk fonksiyonuna uygun olduğu bulunur.

3.3.3. Arama uzayı

Eğer bazı problemleri çözüyorsak bunu diğerleri arasından en iyi çözümü arıyoruz demektir. Mümkün tüm çözümlerin uzayına(istenen çözümün aralarından bulunduğu çözümler kümesi) arama uzayı(arama genişliği) olarak adlandırılır. Bu arama genişliğindeki her nokta mümkün olan bir çözümü gösterir. Her mümkün çözüm değeri ve problem için uygunluğu ile seçilir(işaretlenir). Olası çözümler arasındaki bir nokta(ya da daha fazla) olan çözümü arıyoruz. Buradaki nokta arama uzayındaki bir noktadır.

Çözüm aramak en uç değerleri(minimum veya maksimum) aramak ile aynı anlamdadır. Zaman zaman arama uzayı iyi tanımlanmış olabilir, ama bu arama uzayında sadece bir kaç noktayı biliyor olabiliriz. GA kullanma sürecinde, çözüm bulma süreci diğer noktaları (olası çözümleri) evrim sürdükçe üretir.



Şekil 3.2. Arama uzayı'nın(genişliği) örneği [5]

Sorun, aramanın çok karmaşık olmasıdır. Çözüm için nerden başlanacağı ve nerde aranacağı bilinmeyebilir. Uygun çözüm(en iyi çözüm olması gerekmez) bulmak için birçok metot vardır.

Bu yöntemlerden bazıları hill climbing (tepe tırmanma), tabu search (yasak arama), simulated annealing (benzetimli tavlama) ve genetik algoritmalardır. Bu metotla bulunan çözümler iyi olarak kabul edilir, çünkü optimumu bulmak ve kanıtlamak genellikle mümkün değildir.

3.3.4. NP-Zor("NP Hard") problemleri

"Geleneksel" yolla çözülemeyen problem sınıfına giren NP-Zor ("Non-deterministic Polynomial Time") problemlerine örnektir.

Hızlı algoritmaların uygulandığı birçok görev vardır. Ancak algoritmik olarak çözülemeyen bazı problemler vardır. Bazı problemlerin polinomsal zamanla çözülemediği kanıtlanmıştır. Çözüm bulmanın çok zor olduğu önemli problemler vardır fakat çözüm bulununca bu çözümü kontrol etmek kolaydır. Bu gerçek "NP-Complete" problemlerini ortaya çıkarır. NP Nondeterministic Polynomial anlamına gelir ve bunun anlamı çözüm (Nondeterministic algoritma yardımıyla) "tahmin" edilebilir ve kontrol edilebilir.

Eğer tahmin edebilen bir makineye sahip olsaydık uygun bir zaman içinde bir çözüm bulmaya çalışılır.

NP Problemlere örnek olarak tatmin problemini, gezgin satıcı problemini(gsp) veya sırt çantası problemini verilebilir.

3.3.5. Popülasyon

Kromozomlardan oluşan topluluğa denir. Popülasyon, geçerli alternatif çözüm kümesidir. Popülasyondaki birey sayısı (kromozom) genelde sabit tutulur. GA'da popülasyondaki birey sayısı ile ilgili genel bir kural yoktur. Popülasyondaki kromozom sayısı arttıkça çözüme ulaşma süresi (iterasyon sayısı) azalır.

3.4. Genetik Algoritmanın Operatörleri (İşleçleri)

Genetik algoritma özetinde incelendiği gibi çaprazlama ve mutasyon GA'nın en önemli kısımlarındandır. Kullanılan operatörler, var olan nesil üzerine uygulanan işlemlerdir. Bu işlemlerin amacı, daha iyi özelliğe sahip nesiller üretmek ve arama algoritmasını genişletmektir ve genetik algoritmanın performansını oldukça yüksek bir düzeyde etkilemektedir. 3 tip genetik operatör vardır.

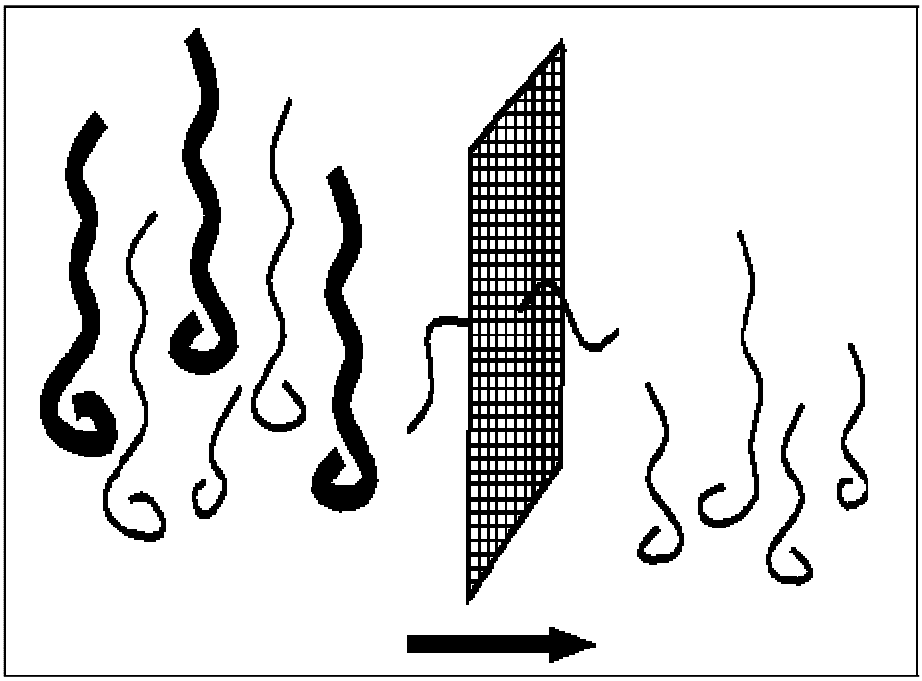
- Seleksiyon(Seçilim)
- Çaprazlama
- Mutasyon

3.4.1. Seleksiyon (Seçilim)

Kromozomlar, çaprazlama için popülasyondaki ebeveynlerden seçilir. Problem kromozomların nasıl seçileceğidir. Darvin 'in evrim teorisine göre en iyi olanlar hayatta kalır ve yeni nesiller oluştururlar. Yeniden üretme operatörü, hazır topluluktan uygun olan bireylerin seçilmesi ve bunların sonraki topluluğa kopyalanarak hayatta kalmalarıyla ilgilidir. Seçim modeli, tabiatın hayatta kalabilmek için uygunluk mekanizması modelidir.

Yeniden üretme işleminde, bireyler onların uygunluk fonksiyonlarına göre kopya edilirler. Uygunluk fonksiyonu, mümkün olduğu kadar yükseltilmesi gereken bazı

faydalı ve iyi ölçülerdir. Topluluk uzayındaki her bir bireyin uygunlukları baz alınarak ne kadar sayıda kopyasının olacağına karar verilir. En iyi bireylerden daha fazla kopya alınır, en kötü bireylerden kopya alınmaz. Bu hayatta kalmak için uygunluk stratejisinin GA ya sağladığı avantajdır.



Şekil 3.3. Neslin devam etmesi için bireyler arasından elemeye(süzme) devam eder.

En iyi kromozomu seçmek için bazı yöntemler vardır. Örneğin; rulet tekerlek seçimi(roulette whell selection), boltzman seçimi, turnuvalı seçim, sıralı seçim(rank selection), durgun durum veya kararlı hal(steady-state) seçimi vb.

— **Rulet Tekerliği Seçilimi(Roulette Whell Selection)**

Ebeveynler uygunluklarına göre seçilir. Daha iyi kromozomlar seçilmek için daha fazla şansa sahiptirler. Tekrar üretim başlangıcında basit bir yöntem olan rulet tekerliği seçimine göre popülasyondaki bütün kromozomların uygunluk değerlerine göre rulet tekerleği hazırlanır.

Bilye(marble) rulet tekerleğine atılmakta bilyenin durduğu yerdeki kromozom seçilir. Uygunluğu fazla olan kromozomlar daha fazla seçilir.

Seçilecek bireyi belirlemek için, öncelikle eşlenik 3.1 deki formül ile uygunluk değerlerinin toplamı bulunur.

$$UT = \sum_{i=1}^{PS} F_i \qquad (3.1)$$

Burada **PS** popülasyondaki birey sayısı (popülasyon büyüklüğü), U_i ise her bir bireyin uygunluk değeridir. Elde edilen toplam uygunluk değeri kullanılarak her bir bireyin seçilme olasılığı (O_i) yüzde olarak Eşlenik 3.2 deki formül ile hesaplanır.

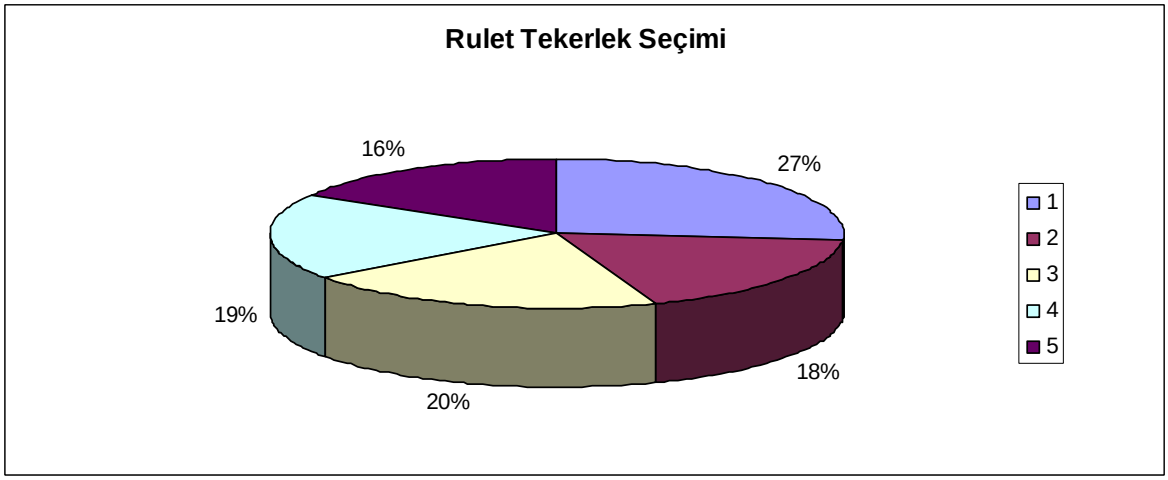
$$O_i = U_i / UT * 100 \qquad (3.2)$$

Örnek; Her x değeri için F(x)(255–2*x) fonksiyonu uygulanır. Bu her kromozomun fonksiyon değerine göre uygunlukları (2-f(x)/∑f(x)) belirlenir. Yukarıdaki 3.1 ve 3.2 deki eşleniklere göre tüm kromozomların uygunluk değerleri toplanır ve her kromozomun seçilme olasılığı hesaplanır.

Tablo 3. 1. Seçilen bireyler ve değerleri

Birey No	x	F(x)	Uygunluk	Seçilme Olasılıkları
1	104	47	1,62	26,40833019
2	191	127	1,11	18,15813132
3	72	111	1,26	20,4389419
4	56	143	1,15	18,70902255
5	224	193	1,00	16,28557403

Hesaplanan olasılık değerlerinin sonucu rulet tekerleğinde aşağıdaki şekilde görüldüğü gibi gösterilebilir.



Şekil 3. 4 Rulet Tekerlek Seçimi

Süreç aşağıdaki algoritma ile anlatılabilir.

1 [Toplam]: Popülasyondaki tüm kromozomların uygunluk toplamını hesaplanması -**S**.

2 [Seçim]: (0,S) aralığından rasgele bir sayı üretilir - **r.**

3 [Döngü]: Toplum üzerinden gidip 0'dan itibaren uygunlukların toplamını al - s, s r'den büyük olduğu zaman dur ve bulunduğumuz yerdeki kromozomu döndür.

***1.aşama her popülasyonda bir kere gerçekleşir.*

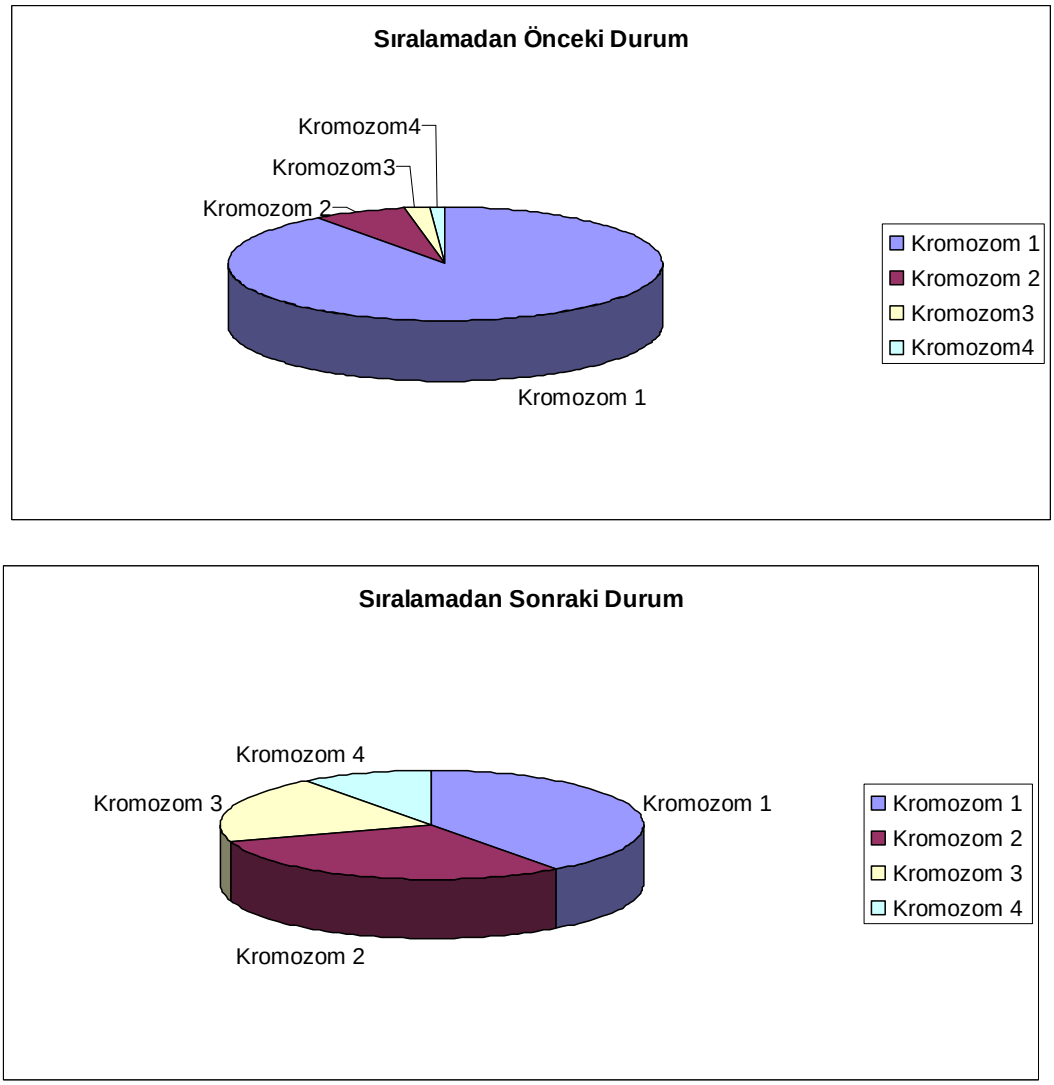
— **Sıralı Seçilim (Rank Selection)**

Bir önceki seçim düzeneğinde uygunluk değerleri arasında büyük farklar oluşunca problemler ortaya çıkacaktır.

Örneğin; eğer en iyi kromozomun uygunluğu diğer tüm kromozomların toplamının %90'ı ise diğer kromozomların seçilme sansı çok azalacaktır. Sıralama seçimi ilk önce toplumu sıralar ve her kromozom uygunluk değeri olarak sırasını kullanır. En

kötü 1 uygunluğunu, ikinci kötü 2,…., en iyi N (toplumdaki kromozom sayısı) uygunluğunu alır.

Aşağıdaki grafiklerde, sıra numarasına göre uygunluğunun nasıl değiştiği görülür.



Şekil 3. 5. Sıralama seçiminden önceki ve sonraki durum

Bu şekilde tüm kromozomların seçilme şansı olacaktır. Ancak bu yöntem daha yavaş yakınsama neden olabilir, çünkü en iyi kromozomlar birbirlerinden çok farklı değillerdir.

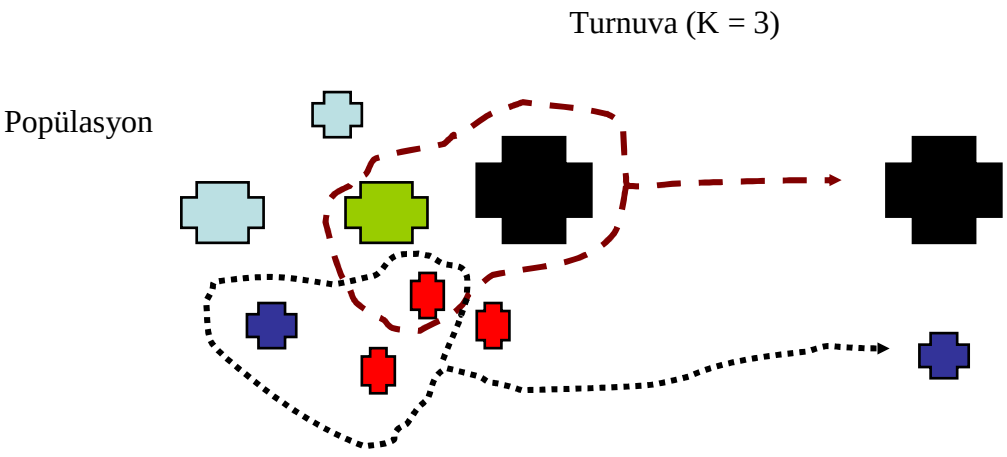
— **Sabit Durum Seçimi(Steady-State Selection)**

Sabit durum seçimi yerine geçme yöntemleri olarak da adlandırılabilir. Bu ebeveynleri seçmek için bir metot değildir. Bu seçimin ana fikri kromozomların büyük bir kısmı bir sonraki nesilde hayatta kalmak zorundadır.

Sabit durum seçimi su şekilde çalışmaktadır. Her yeni nesilde yüksek uygunluk değerine sahip kromozomlar yeni yavruları oluşturmak için seçilir ve düşük uygunluk değerine sahip yavrular kaldırılarak yerlerine bu yeni oluşturulan yavrular koyulur. Toplumun geri kalan kısmı aynen yeni nesle aktarılır.

— **Turnuva Seçimi (Tournament Selection)**

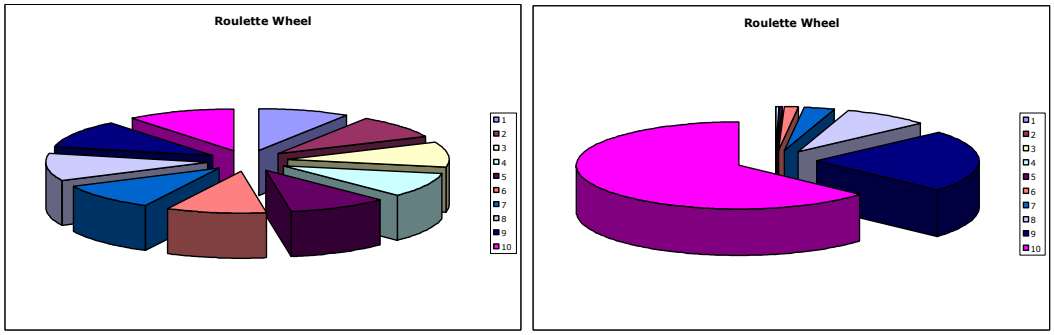
Öncelikle Popülasyon içerisinde rastgele K adet (3,5,7) birey seçilir. Turnuva seçiminde K'nın seçimi genellikle popülasyonun büyüklüğüne göre değişir. Örneğin 10000 büyüklüğünde bir popülasyonunuz varsa o zaman K'yı daha yüksek almalısınız. K=7 ya da daha fazla olmalı. Bu bireylerin içerisinde uyumluluğu (fitness) en iyi olan birey seçilir. Bu işlem seçilen bireylerin sayısı popülasyon sayısına eşit olana kadar devam eder.



Şekil 3. 6. Turnuva (tournament) seçiminin gösterimi

— **Boltzman seçimi**

Bu tür seçimde bireylerin seçilme olasılıklarını hesaplarken sıcaklıklar kullanılır. Eğer bir bireyin uyumluluğu diğerlerinden çok daha baskın ise o popülasyonun sıcaklığı düşüktür. Eğer popülasyondaki bireylerin uyumlulukları eşit oranda dağılıyorsa o zaman o popülasyonun sıcaklığı yüksektir.



Sıcaklık(T) = 50 Sıcaklık(t) = 1

Şekil 3. 7. Boltzman seçimi

— **Elitizm(Seçkinlik)**

Seçkinliğin ana fikri daha önce açıklandı. Çaprazlama ve mutasyon yöntemleriyle yeni bir nesil oluştururken, en iyi kromozomları kaybetme olasılığımız vardır.
Seçkinlik, en iyi kromozomların (ya da bir kısmının) ilk önce kopyalanıp yeni nesle aktarıldığı yöntemin adıdır. Geri kalan kromozomlar yukarıda anlatılan yöntemlerle üretilir.

Seçkinlik GA'nın başarımını hızlı bir şekilde arttırabilir, çünkü bulunan en iyi çözümün kaybolmasını önler.

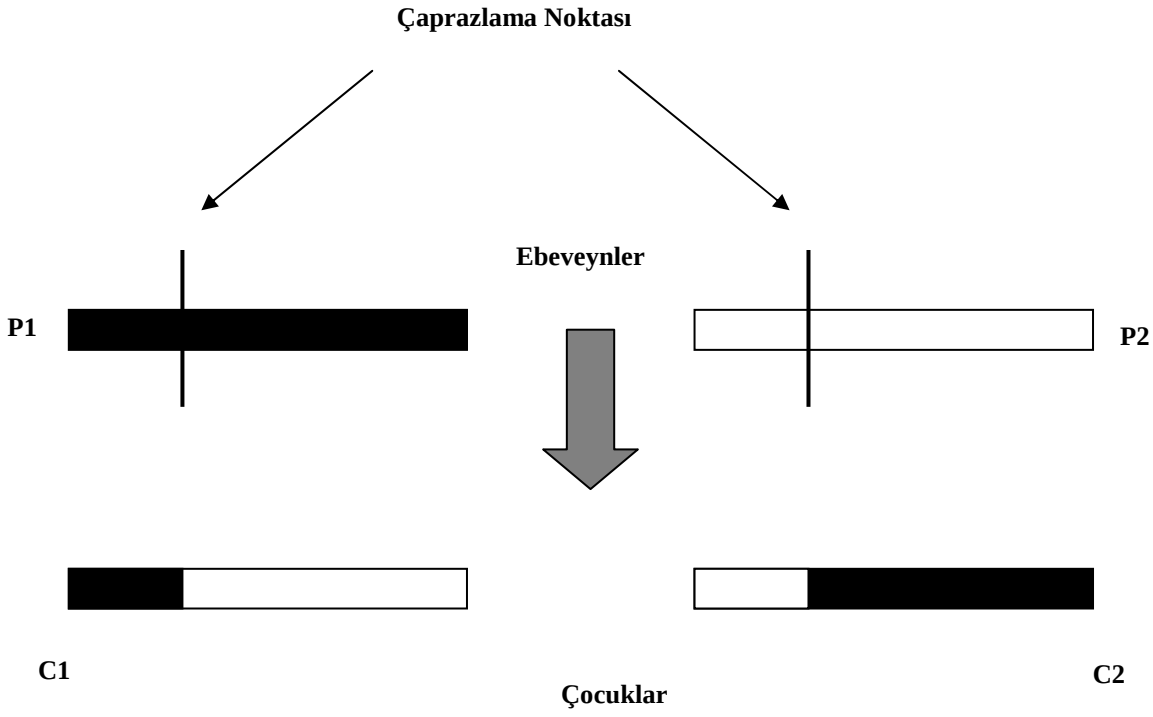
3.4.2. Çaprazlama

Hangi kodlama yöntemi kullanılacağına karar verildikten sonra, çaprazlama işlemine geçilir. Çaprazlama, biyolojik terim olarak ele alınırsa üreme kromozomlarının birbirleriyle yapmış oldukları gen alışverişidir.

Tablo 3. 2. Biyolojik çaprazlama örneği

1.ebeveyn	XX	YY	2.ebeveyn	xx	yy
1.nesil	XXYY		2.nesil	xxyy	

Çaprazlama arama uzayında(çözüm havuzunda) bulunan kromozomları ikişer ikişer birleştirerek yeni çözümler meydana getirmektir. Genetik algoritmada seçim sonrası belli bir olasılıkla kromozomların ikili olarak seçilmesi ve rasgele belirlenen bir noktanın sağ tarafındaki genlerin karşılıklı olarak değiştirilmesi sonucu çaprazlama işlemi gerçekleştirilmiş olur. İki kromozomdan iki adet yeni kromozom üretilmektedir. Bir problem çözüm uzayında kaç adet kromozom un çaprazlanacağı önceden belirlenen çaprazlama oranına göre belirlenmektedir.



Şekil 3. 8. P1 ve P2 ebeveynin çaprazlanmasıyla iki çocuk(nesil) oluşur. Bunlar C1 ve C2 diye adlandırılan nesillerdir. Her ikisi ebeveynlerden birer parça almıştır. Tek noktalı çaprazlamaya örnektir.

Çaprazlamanın amacı, ebeveynlerin kromozomlarının genleri değiştirerek çocuk (child, yavru, nesil) kromozomlar üretmek ve böylece var olan uygunluk değerini yüksek olan kromozomlardan, daha yüksek uygunluk değerine sahip olanları

üretmektir. Fakat hangi özelliklerin iyi performans sağladığına yönelik bir fikir edinilmediği için özelliklerin değiş tokuşu şeklinde birleşim rassal olarak gerçekleştirilir. Bu şekilde rassal olarak yapılan birleşimler ile iyi sonuçlar alınması beklenir. Tabi ki bazen en kötü özelliklerin toplandığı bir çocuk oluşumu da söz konusu olabilir. Bu durumda bu çocuk elenecektir.

Ele alınan probleme bağlı olarak, kullanıcı tarafından seçilen 4 farklı çaprazlama operatörü bulunmaktadır.

- Tek nokta çaprazlama
- İki nokta çaprazlama
- Çok nokta çaprazlama
- Üniform Çaprazlama

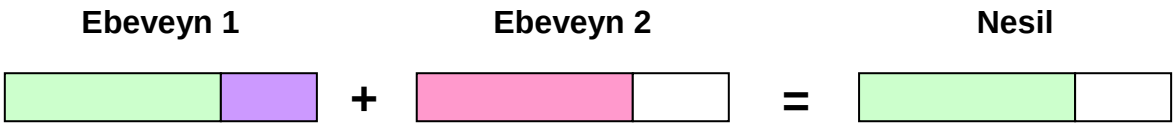
Tek nokta çaprazlama, genetik algoritmanın kullandığı en basit çaprazlamadır. Rastgele seçilen kromozom çiftine çaprazlama uygulanır. Tek nokta çaprazlama işlemi için kromozomda çaprazlama yapılacak bölge kullanıcı tarafından rastgele seçilebilir. Oluşan yeni birey ebeveynlerin bazı özelliklerini alarak her ikisinin kopyası olacaktır.

K1=10101011

K2=11101000

} **C1**=10101000 ve **C2**=11101011

Açıklamalar;

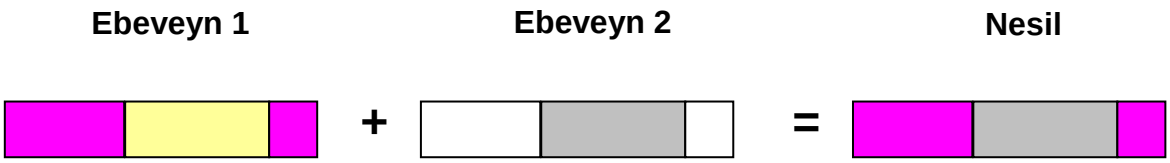


İki nokta çaprazlamada iki nokta arasında kalan alt dizilerin değiştirilmesiyle iki yeni birey elde edilir.

K1=00110110

K2=11101000

C1=00101010

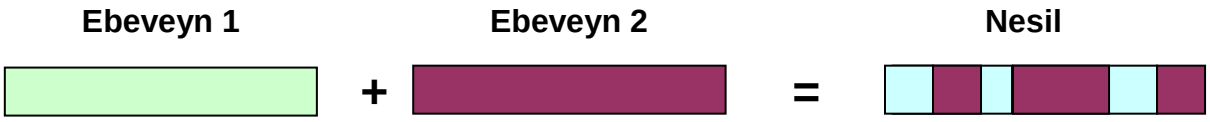


Çok nokta çaprazlama yöntemi, iki nokta çaprazlamanın gelişmiş bir halidir. Kromozomlar daha fazla parçalara ayrılır ve bir atlanarak elde edilen çiftler arasında değiştirilerek yeni bireyler elde edilir.

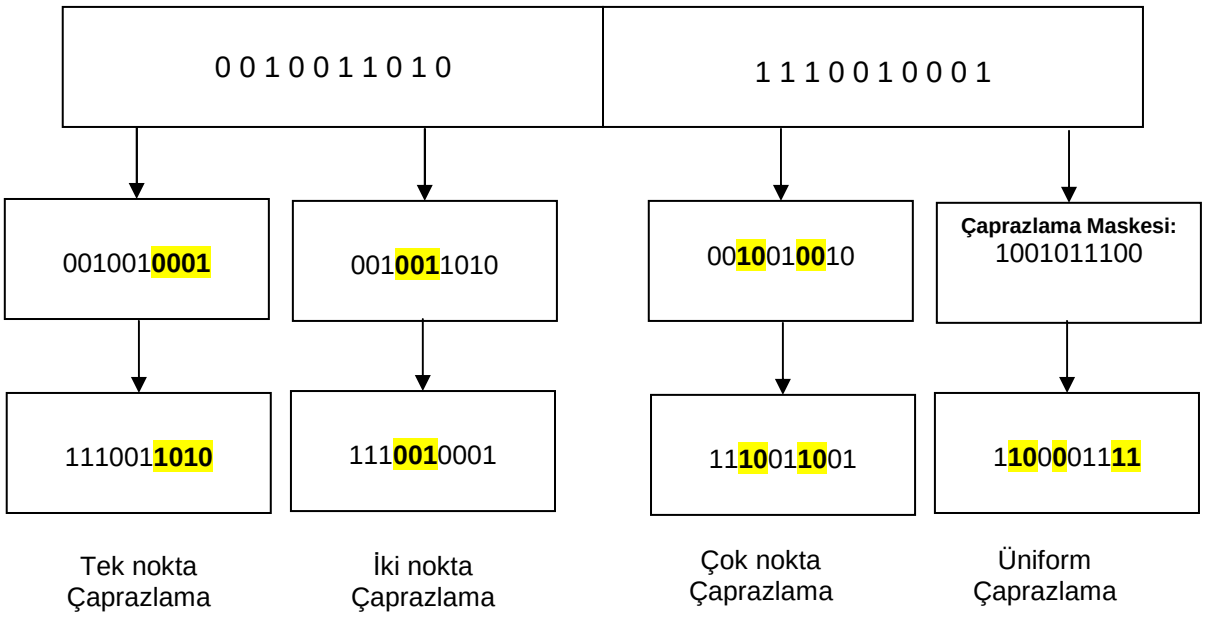
K1=10101011

K2=01010000

C1=01011100



Üniform çaprazlamada rastgele olarak çaprazlama maskesi oluşturulur. Birinci ve ikinci kromozoma karşılık gelen genin kopyalanmasıyla yaratılır. Çaprazlama maskesinde bir o genin birinci kromozomdan, sıfır ise o genin ikinci kromozomda kopyalanacağı anlamına gelmektedir.



Şekil 3. 9. Çaprazlama yöntemleri ve etkileri

İki kromozomdan çaprazlama yapılmış elemanlar, Şekil 3.9 de her bir çaprazlama operatörü için sarı renkte gösterilmiştir.

Çaprazlamadan başka tersinme denilen bir üreme yöntemi daha vardır. Holland bunu tanımlayarak kromozom uzunluğu çok olan bireylerde çaprazlama yerine bunun kullanılmasını performans açısından önermiştir. Tersinme (inversion) bir kromozomu oluşturan genlerden ardışık bir grubun kendi içerisinde birbirleriyle yer değiştirerek ters dizilmeleridir.

Örneğin: 10101000111 kromozomu(her genin bir bit olduğu varsayımı ile) 6. Ve 9. Gen kromozomları arasında tersindiğinde ortaya 10101100011 kromozomu çıkar. Tersinme genellikle kromozom uzunluğu fazla olan popülasyonlara uygulanır.

3.4.3. Mutasyon

Çaprazlama işlemi gerçekleştirildikten sonra, mutasyon işlemi gerçekleşir. Mutasyonun amacı popülasyondaki(toplumdaki) bütün çözümlerin çözülen problemlerin yerel optimumlarına düşmesi engellemektir.
Diğer amaç ise; var olan bir kromozomun genlerinin bir ya da birkaçının yerlerini değiştirerek yeni kromozom oluşturmaktır. Yeniden ve sürekli yeni nesil üretimi sonucunda belirli bir süre sonra nesildeki kromozomlar birbirlerini tekrarlama

konumuna gelebilir ve bunun sonucunda farklı kromozom üretimi durabilir veya çok azalabilir. İşte bu nedenle nesildeki kromozomlarının çeşitliğini artırmak için kromozomlardan bazıları mutasyona uğratılır. Açıklandığı gibi mutasyonun birinci maksadı bir popülasyonun içindeki değişimi tanımlamaktır.

Mutasyon popülasyonlarda çok önemlidir. Öyle ki burada ilk popülasyon mümkün olan tüm alt çözümlerin küçük bir alt kümesi olabilir ve ilk popülasyondaki tüm kromozomların önemli biti sıfır olabilir. Hâlbuki o bitin problemin çözümü için 1 olması gerekebilir ve bunu da çaprazlama düzeltemeyebilir. Bu durumda o bit için mutasyon kaçınılmazdır.

Genellikle kullanılan mutasyon oranı, birim birey gen uzunluğuna bölümü seviyesindedir. Örneğin 100 gen birimine sahip bir birey için oran 0.01' dir. Diğer deyişle rassal olarak düşünüldüğünde her bir genin mutasyona uğrama olasılığı %1'dir. Bu işlem çaprazlamadan sonra gelir. Mutasyonun yapılıp yapılmayacağını bir olasılık testi belirler. Örneğin yeni neslin ortalama uygunluğu ≤ Eski neslin ortalama uygunluğu ise; x. Kromozomun y. Bitini değiştir denilebilir. Bu yeni çocuğu rast gele değiştirir. İkili kodlama için rast gele seçilmiş bitlerden 0'ları 1, 1'leri 0 yaparız.

Mutasyon işleminin uygulanmasında gözden kaçırılmaması gereken başka bir konu ise, kromozomun bünyesindeki genlere uygulanacak değişikliğin sonucunun, genin kromozom içerisindeki konumuna bağlı olduğudur. Diğer bir ifade ile kromozomun genlerinin birinci elemanına uygulanacak bir değişiklik, son elemanına uygulanacak bir değişiklikten çok daha belirgindir. 8 elemanlı ikili sayıdan oluşan bir bromozomun birinci elemanında yapılacak bir değişikliğin ondalık karşılığı 128 iken (2^7), 8 nci elemanında yapılacak değişikliğin ondalık karşılığı 1'dir. Bu sebeple, iterasyonun sonuna yaklaştıkça daha küçük değişikliğe sebep olacak mutasyona izin verilmesi uygun görülmektedir.

Mutasyonun her adımda her yeni bireye rastsal olarak uygulanması yerine farklı uygulamalar da mevcuttur. Hacıoğlu ve ark., iterasyonun sadece belli adımlarında tüm bireylerin tüm genlerine mutasyon uygulamışlardır. Aradaki adımlarda mutasyon uygulamayarak çözümün kendisini toparlamasına izin vermişlerdir. Uygulanacak titreşimin genliğini, ortalama uygunluk değerinin değişimine bağlı olarak uyarlamışlardır. Titreşimli genetik algoritma (TGA) olarak adlandırdıkları bu

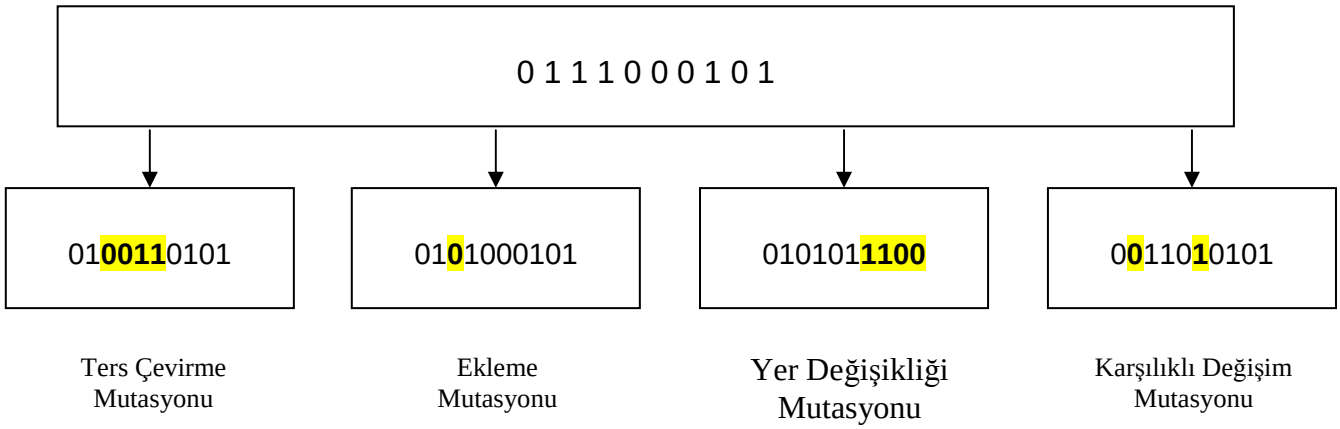
yöntemin, geleneksel mutasyon uygulanan yönteme göre daha başarılı olduğu belirtilmiştir.

Ele alınan problemin yapısına göre en uygun olan aşağıdaki mutasyon çeşitlerinden biri seçilir.

- Ters çevirme
- Ekleme
- Yer değişikliği
- Karşılıklı Değişim

Mutasyon operatörlerinin uygulamaları Tablo 2.4'te görülmektedir. Şekilde sarı renkle gösterilen eleman değerleri mutasyona uğramış elemanları göstermektedir.

- Ters çevirmede, kromozomdan rastgele iki pozisyon seçilir ve iki ucu arasında ters çevrilir.
- Eklemede ise rastgele bir parka(bit) seçilir ve rastgele bir yere yerleştirilir.
- Yer değişikliği mutasyonunda, rastgele bir alt dizi seçilir ve rastgele bir yere yerleştirilir.
- Karşılıklı değişim mutasyonunda, rastgele seçilen iki genin yerleri değiştirilir.



Şekil 3. 10. Mutasyon yöntemleri ve etkileri

3.5. GA Parametreleri

GA’nın iki parametresi da vardır. Bunlar çaprazlama ve mutasyon ihtimalleridir.

— **Çaprazlama İhtimali**

Çaprazlamanın ne kadar sıklıkla olduğunu gösterir. Eğer hiç çaprazlama yoksa nesil ebeveynlerin kopyası olur. Eğer çaprazlama varsa nesiller ebeveynlerinin kromozomlarının parçalarından oluşur. Çaprazlama olasılığı %100 ise bütün oluşan nesil(çocuk, döl) tamamen çaprazlamayla oluşmuştur. Eğer çaprazlama olasılığı %0 ise bütün nesil eski popülasyonun kromozomlarının aynılarını taşır. (Fakat bu yeni neslin eski neslin aynısı olduğu anlamına gelmez.)
Çaprazlama yeni kromozomların eski kromozomlardan iyi parçaları aldığı beklenir. Belki de yeni kromozomların daha iyi olması beklenir. Bununla birlikte popülasyonun gelecek nesli devam etmesi için bazı kromozomların gelecek nesle bırakılması gerekir.

— **Mutasyon İhtimali**

Kromozomların hangi sıklıkla mutasyon geçirdiğini gösterir. Eğer mutasyon yoksa çaprazlamadan sonra nesil değişmeksizin alınır. Eğer mutasyon varsa kromozomun bir parçası değişir.
Mutasyon ihtimali %100 ise bütün kromozomlar değişir. Eğer mutasyon ihtimali %0 ise kromozom hiçbir değişikliğe uğramaz. Mutasyon GA’nın yerel minimum ve maksimum noktalarda bulunmasını engeller fakat bu çok sık gerçeklenmez çünkü genetik algoritma rasgele aramayla değişir.

3.6. Diğer Parametreler

GA’nın diğer önemli parametresi ise popülasyonun büyüklüğüdür.

3.6.1. Popülasyon büyüklüğü

Popülasyondaki kromozom sayısını verir. Bir nesilde eğer az kromozom varsa GA da çaprazlama ihtimali düşer ve sadece arama uzayının bir kısmını tarar. Diğer yönden çok fazla kromozom varsa GA yavaşlar. Araştırma limitten sonrasını gösteriri.(Bu popülasyonun artmasında kullanılmaz çünkü problemin hızlı çözülmesini sağlanmaz.)

3.6.2. Kodlama

Kromozomların kodlanması bir problem çözümüne başlanırken sorulması gereken ilk sorudur. Kodlama problemin kendisine yoğun şekilde bağlıdır.

— **İkili kodlama**

İkili kodlama en çok kullanılan yöntemdir, çünkü ilk GA araştırmaları bu kodlama yöntemini kullanıldı ve görece basit bir yöntemdir. İkili kodlamada, her kromozom bit (0 veya 1) karakter dizilerinden oluşmaktadır.

Tablo 3. 3. İkili Kodlanmış Kromozom Örnekleri

Kromozom 1	0 1 1 1 1 0 1 0 0 1 0 1 0 1 1 0 1 1 1 0 0 1 0 0
Kromozom 2	1 0 0 0 1 0 1 1 0 1 0 1 0 1 0 0 0 0 0 1 1 1 1 0

İkili kodlama, fazla olasılıkta kromozomlar verir, bunlara düşük sayıda alel içerenler de dahildir. Ancak, bu yöntem çoğu problem için doğal bir kodlama değildir ve çaprazlama ve/ya mutasyondan sonra düzeltmeler yapılması gerekir.

— **Permütasyon Kodlama**

Permütasyon kodlama, gezgin satıcı problemi veya görev sıralama gibi sıralama problemlerinde kullanılabilir. Permütasyon kodlamada, her kromozom sıra da konum belirten numara karakter dizisinden oluşur.

Tablo 3. 4. Permütasyon kodlanmış kromozom örnekleri

Kromozom 1	9 7 8 6 1 2 3 5 4
Kromozom 2	7 8 9 2 1 6 3 4 5

Permütasyon kodlama, sıralama problemleri için yararlıdır. Bazı problemlerde bazı çaprazlama ve mutasyon türleri için kromozomların tutarlılığı için (örneğin içerisinde gerçek sırayı tutan) düzeltmeler yapılması gerekmektedir.

— **Tek Noktalı Çaprazlama**

Bir kesme noktası seçilir, kesme noktasına kadar ilk atadan, kesme noktasından sonraki kısımlar da ikinci atadan olmak üzere permütasyonlar kopyalanır. Aynı sayılar olmayan sayılarla değiştirilerek tutarlı yeni yavru elde edilir. Bundan çok farklı, daha fazla sayıda yöntem de uygulanabilir.

Ebeveyn 1= 1 2 3 4 5 6 7 8 9

Ebeveyn 2= 3 5 4 9 8 7 6 1 2

Nesil=1 2 3 4 5 9 8 7 6

1) Mutasyon Yöntemi (Yer Değiştirme)

İki sayı seçilir ve yerleri değiştirilir.

Tablo 3. 5. Mutasyon Yöntemi

Mutasyondan Önce	1	2	3	4	5	6	7	8	9
Mutasyondan Sonra	1	2	**9**	4	5	6	7	8	**3**

değişime uğrayan genler

— **Değer Kodlama**

Gerçek sayılar gibi karmaşık değerlerin kullanıldığı problemlerde doğrudan değer kodlama kullanılabilir. İkili kodlamanın bu tip problemler için kullanılması problemlerin zorlaşmasına neden olacaktır. Değer kodlamada, her kromozom bazı değerlere eşittir. Değerler problemle ilgili herhangi bir şey olabilir. Gerçek sayılar, karakterler veya herhangi nesneler olabilir.

Değer kodlama bazı özel problemler için iyi bir seçimdir. Ancak, bu tip kodlamada probleme özgü yeni çaprazlama ve mutasyon yöntemleri geliştirmek gereklidir.

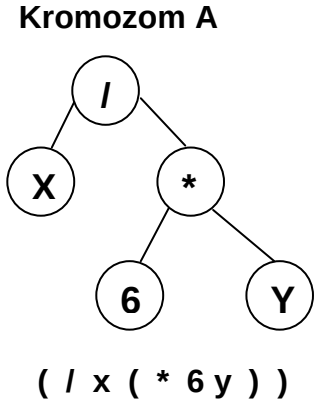
Tablo 3. 6. Değer kodlama ile kodlanmış kromozom örnekleri

Kromozom 1	0.3245 – 1.2345 – 5.6789 – 2.3256 – 2.1214
Kromozom 2	AERTYUIOPZCVXBNMSDFGHJKL
Kromozom 3	Doğu – Batı – Kuzey - Güney

a) Ağaç Kodlama

Ağaç kodlama genellikle gelişen, değişen program veya ifadeler için kullanılmaktadır. Örnek olarak genetik algoritmayı verebiliriz.

Ağaç kodlamada her kromozom bazı nesnelerin, mesela fonksiyonlar ya da programlama dilindeki komutlar gibi, bir ağaçtır. LISP programlama dilinde programların ağaç seklinde temsil edilmesi nedeniyle LISP bu iş için en çok kullanılan dildir. LISP'te bu ağaçlar kolayca ayrıştırılıp, çaprazlama ve mutasyon kolayca yapılma



Şekil 3. 11. Ağaç kodlamada kromozom örneği

Kromozomlar ağaç kodlama yöntemiyle kodlandıktan sonra ağaç kodlama da çaprazlama işlemi gerçekleştirilir.

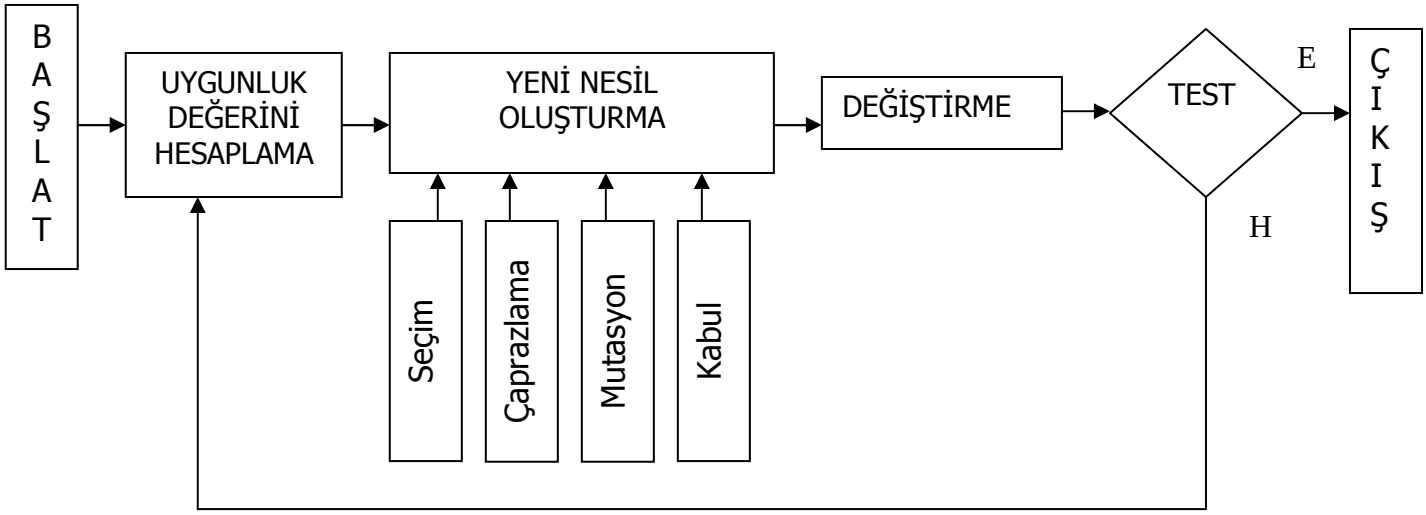
3.7. Genetik Algoritmaların Çalışma Prensibi

Genetik algoritmalar doğal seçim ilkesine dayanan bir sayısal optimizasyon yöntemidir. Genetik algoritma, çözüm dizilerinden oluşan bir başlangıç nesliyle, çaprazlama ve mutasyon gibi doğal seçim operatörlerini kullanmaktadır.

Genetik algoritmalar oldukça genel prensiplerle Şekil 3.12'de akış şemasında görüldüğü gibi çalışmaktadır. Öncelikle ele alınan problem için bir rastgele *n* kromozomlu popülasyon oluşturulur. Daha sonra popülasyondaki her bir kromozom için *f(x)* uygunluk fonksiyonu hesaplanır. Yeni bir popülasyon oluşuncaya kadar aşağıdaki adımlar tekrar edilir:

1. **[Başlangıç]:** n kromozom oluşan rasgele bir popülasyon oluşturulur (problemin olası çözümleri)
2. **[Uygunluk]:** Popülasyondaki her bir kromozom olan x'in f(x) fonksiyonunda değerlendirilir.
3**. [Yeni Popülasyon]:** Yeni popülasyon tamamlana kadar aşağıdaki adımların tekrar edilmesiyle yeni popülasyon oluşturulur.
    a. **[Seçilim(Seleksiyon)]:** Popülasyondan iki ebeveyn kromozom seçilir.(daha uygun olanın seçilme sansı daha fazladır)
    b**. [Çaprazlama]:** Çaprazlama olasılığıyla yeni nesil oluşturmak için eşleştirilir. Eğer hiç çaprazlama olmazsa çocuklar ebeveynlerinin aynıları olur.
    c. **[Mutasyon]:** Mutasyon olasılığı ile her locustaki yeni nesil'i mutasyona uğratır.
    d**. [Kabul etme]:** Yeni yavru(nesil), yeni popülasyona eklenir.
4. **[Yer Değiştirme]:** Yeni toplum algoritmanın tekrar islenmesinde kullanılır.
5**. [Test]:** Eğer son durum yeterliyse dur ve çalışan popülasyondaki en iyi sonuca geri döner.
6. **[Döngü]:** 2. adıma geri dönülür.

Yeni popülasyon kabul edildikten sonra, oluşturulan yeni populasyon eskileriyle yer değiştirilir. Hedeflenen uygunluk değerine ulaşıldığında program durdurulur ve popülasyondaki en iyi çözüm alınır.



Şekil 3. 12. Genetik algoritmanın genel akış şeması

Genetik algoritmalarda kromozomlarla bir başlangıç popülasyonu rasgele oluşturulur. Burada popülasyon genişliğinin belirlenmesi gerekmektedir. Büyük popülasyonlarda, çözüm uzayı iyi örneklendiği için aramanın etkinliği artmakta, fakat buna bağlı olarak da arama süresi uzamaktadır. Küçük popülasyonlarda ise, çözüm uzayını yeterli örnekleyememe ve zamansız yakınsama oluşabilmektedir.

Genetik algoritmanın her çevriminde, yığındaki dizilerin bir değerlendirme fonksiyonu yardımıyla uygunluk değeri hesaplanır. Uygunluk fonksiyonu, kromozomları problemin parametreleri haline getirmekte ve bunlara göre hesaplama yapmaktadır. Genellikle genetik algoritmaların başarısı bu fonksiyonun verimli ve hassas olmasına bağlıdır.

Yukarıda da görüldüğü gibi, genetik algoritmanın akışı oldukça kolaydır. Birçok parametre ve ayar farklı problemler için farklı şekillerde gerçekleştirme için vardır. Sorulması sorulan ilk soru kromozomun nasıl kodlanacağıdır. Daha sonra çaprazlama ve mutasyon, GA'nın iki basit operatörü adreslenecektir.

Bir sonraki soru çaprazlama için ataların nasıl seçileceğidir. Bu farklı birçok yolla yapılabilir, ancak ana fikir daha iyi ataların daha iyi yavrular üreteceği düşüncesiyle seçilmesidir. Bu şekilde en iyi çözümün kaybedilmemesi için seçkinlik, en iyi çözümün değiştirilmeden yeni nesle aktarılması, böylece en iyi çözümün yaşatılması uygulanabilir.

İşlemleri adım adım açıklamak gerekirse;

Adım–1 Bu adımda toplumda bulunacak birey sayısı belirleyerek başlanmaktadır. Kullanılacak sayı için bir standart yoktur. Genel olarak önerilen 100–300 aralığında bir büyüklüktür. Büyüklük seçiminde yapılan işlemlerin karmaşıklığı ve aramanın derinliği önemlidir. Toplum bu işlemden sonra rasgele oluşturulur.

Adım–2 Kromozomların ne kadar iyi olduğunu bulan fonksiyona uygunluk fonksiyonu denir. Bu fonksiyon işletilerek kromozomlarının bulunmasına ise hesaplama denir. Bu fonksiyon genetik algoritmanın temelini oluşturmaktadır. Genetik algoritmada probleme özel çalışan tek kısım bu fonksiyondur. Uygunluk fonksiyonu kromozomları problemin parametreleri haline getirerek onların bir bakıma şifresini çözmektedir. Sonra bu parametrelere göre hesaplamayı yaparak kromozomların uygunlukları belirlenir. Çoğu zaman genetik algoritmanın başarısı bu fonksiyonun verimli ve hassas olmasına bağlı olmaktadır.

Adım–3 Kromozomların eşlenmesi kromozomların uygunluk değerine göre yapılır. Bu seçimi yapmak için rulet tekerleği seçimi, turnuva seçimi gibi seçme yöntemleri vardır. Seçilim yapıldıktan sonra çaprazlama, mutasyon gibi işlemlerden geçerek yeni kromozomlar üretilir.

Adım–4 Eski kromozomlar çıkartılarak sabit büyüklükte bir toplum sağlanır.

Adım–5 Tüm kromozomlar yeniden hesaplanarak yeni toplumun başarısı bulunur.

Adım–6 Genetik algoritma defalarca çalıştırılarak çok sayıda toplum oluşturulup hesaplanır.

Adım–7 Toplumların hesaplanması sırasında en iyi bireyler saklandığı için o ana kadar bulunmuş en iyi çözümdür.

3.8. Genetik Algoritmanın(GA) Avantajları Ve Dezavantajları

Genetik algoritma ile Eİ sorunlarının çözülmesindeki bazı avantajlar;

1) Kesikli veya sürekli değişkenlerle Eİ yapılabilir.
2) Türev alma işlemine gerek yoktur.
3) Çözüm uzayında aynı anda geniş bir alanda çok sayıda noktadan araştırmaya başlanır.
4) Çok fazla sayıda değişkenle Eİ işlemleri yapılabilir.
5) Paralel hesaplamalara çok uygundur.
6) Çok fazla uç(en büyük ve en küçük) değeri olan hedef fonksiyonları durumunda bile Eİ yapılabilir.
7) Yerel en küçüklemeye sıçrayarak aşabilir.
8) Sadece mutlak en iyi çözümü değil en iyi çözümlemelerin listesini bile verebilir.
9) Karar değişkenlerini kodlayarak Eİ'yi yi kodlama dünyasında yapar.
10) Genetik sayı sistemine göre üretilen sayılarla çalışır. Bunlar deney verileri veya analitik fonksiyonlar olabilir.
11) GA parametrelerin kodlarıyla uğraşır.
12) Genetik algoritma ne yaptığı konusunda bilgi içermez, nasıl yaptığını bilir. Bu nedenle kör arama metodudur.

GA'nın dezavantajlarından biri en iyi çözümün bilinen çözümler arasında göreceli olmasıdır. Ulaşılan çözümün en iyi olup olmadığının kontrol edilmesine meydan vermeyebilir. Bu nedenle, GA'la en iyi çözümün kesinlikle ne olabileceğinin bilinmemesi durumunda kullanılır. Doğal olarak genetik algoritmaları en iyi çözüm için ne zaman duracaklarını bilemezler. Bu durma

kriterleri belirlenmelidir. Klasik Eİ yöntemlerinde bir yön veya sistematik izin takip edilmesi gerekmesine karşı, GA karar uzayının tamamen rasgele olarak taranacağı belirlenmiştir. Eİ araştırma yönteminin diğerlerine göre bazı üstünlüklerini şöyle sıralayabiliriz;

1) GA karar uzayında aynı anda ve paralel olarak birçok noktada değerlendirme yaparken, bu durum klasik yöntemlerde tek nokta ve yön olarak karşımıza çıkar.
2) GA işlemleri türev ve integral gibi hesaplamalar gerektirmez ve tüm hesaplamalar aritmetik işlemlere dayanır. Karar uzayının taranması sırasında önemli etkiler sadece hedef fonksiyonu ve ona bağlı olarak ortaya çıkan uygunluk dereceleridir.
3) GA işlemleri sırasında belirgin(deterministik) değil ihtimali ve skokastik geçişler kullanılır.
4) GA'da değişkenler gerçek ondalık değerleri olabilecek çok fazla sayıda çözüm getirir.

Sonuç olarak en iyilemenin

- Bir işin daha iyi yapılması,
- En doğru şekilde yapılması,

olmak üzere iki amacı vardır.

Günümüzde rasgele aramaların kullanımı artmaktadır. Bu tip aramalar en iyilemenin daha iyi yapma amacını sağlamakta daha başarılıdırlar. İnsanların bilgisayarlardan genel beklentisi mükemmellik olduğu için bu tip aramalar başarısız görünebilir. Genetik algoritmalar klasik yöntemlerin çok uzun zamanda yapacakları işlemleri kısa bir zamanda çok net olmasa da yeterli bir doğrulukla yapabilir.

GA'nın Performansını Etkileyen Nedenler

- **Kromozom sayısı**: Kromozom sayısını arttırmak çalışma zamanını arttırırken azaltmak da kromozom çeşitliliğini yok eder.
- **Mutasyon oranı**: Kromozomlar birbirine benzemeye başladığında hala çözüm noktalarının uzağında bulunuyorsa mutasyon işlemi GA'nın sıkıştığı

yerden kurtulmak için tek yoludur. Ancak yüksek bir değer vermek GA'yı kararlı bir noktaya ulaşmaktan alıkoyacaktır.

— **Kaç noktalı çaprazlama yapılacağı**: Normal olarak çaprazlama tek noktada gerçekleştirilmekle beraber yapılan araştırmalar bazı problemlerde çok noktalı çaprazlamanın çok yararlı olduğunu göstermiştir.

— **Çaprazlamanın sonucu elde edilen bireylerin nasıl değerlendirileceği**: Elde edilen iki bireyin birden kullanılıp kullanılamayacağı bazen önemli olmaktadır.

— **Nesillerin birbirinden ayrık olup olmadığı**: Normal olarak her nesil tümüyle bir önceki nesle bağlı olarak yaratılır. Bazı durumlarda yeni nesli eski nesille birlikte yeni neslin o ana kadar elde edilen bireyleri ile yaratmak yararlı olabilir.

— **Parametre kodlanmasının nasıl yapıldığı**: Kodlananın nasıl yapıldığı en önemli noktalardan biridir. Örnek vermek gerekirse kimi zaman bir parametrenin doğrusal ya da logaritmik kodlanması GA'nın performansında önemli bir farka yol açabilir.

— **Kodlama gösteriminin nasıl yapıldığı**: Bu da nasıl olduğu yeterince açık olmamakla beraber GA'nın performansını etkileyen bir noktadır. İkilik düzen, kayan nokta aritmetiği ya da gray kodu ile gösterim en yaygın yöntemlerdir.

— **Başarı değerlendirmesinin nasıl yapıldığı**: akıllıca yazılmamış bir değerlendirme işlevi çalışma zamanını uzatabileceği gibi çözüme hiçbir zaman ulaşmamasına neden olabilir[30].

Örnek: Fonksiyonun En Büyüklenmesi(Maksimizasyon)

Fonksiyonun en büyülenmesi veya en küçüklenmesi GA'ların en fazla kullanıldığı alanlardan birisidir. Aslında türevi alınabilen sürekli fonksiyonların Eİ'si analitik yöntemlerle kolayca yapılabilir.

Amaç: kübik $\mathbf{f(x)=x^3}$ fonksiyonu kullanılacaktır.

Basit bir GA uygulaması için aşağıdaki adımların yapılması gerekir.

1) Kromozomların içinden 6 tane rasgele olarak belirlenen x değeri belirleyelim ve bu x değerlerini ikili tabana(0 ve 1 olarak) çevirelim. Buradan GA başlangıç popülasyonunun 6 kromozomlu seçildiği anlaşılıyor. Her kromozomun uzunlukları 6 gen olarak belirlenir. (Tablo 3.7)

Tablo 3. 7. GA'nın başlangıç popülasyonu

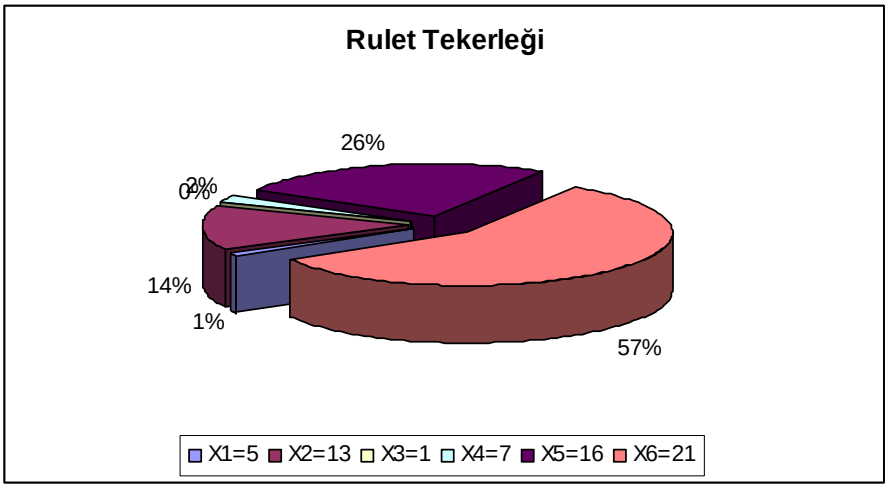
Bireyler	**Kromozomlar**
X_1=5	0 0 0 1 0 1
X_2=13	0 0 1 1 0 1
X_3=1	0 0 0 0 0 1
X_4=7	0 0 0 1 1 1
X_5=16	0 1 0 0 0 0
X_6=21	0 1 0 1 0 1

2) Her kromozomun temsil ettiği X sayılarının fonksiyonda yerleştirilmesiyle bulunan Y sayılarının toplamını sonra da her birinin bu toplam içindeki uygunluk(dinçlik) değerlerini (yüzdelerini) denklemden bulunur. Bu yüzdelerin her biri kromozomların topluluk içinde hayatlarını sürdürme uygunluklarının bir ölçüsüdür. En yüksek uygunluk değerine sahip olan 6. birey en az uygunluk değerine sahip birey ise 3.üncü bireydir.

Tablo 3. 8. GA'nın hedef ve uygunluk değerleri

Bireyler	**Hedef Değerler**	**Uygunluk Değerleri**
X_1=5	125	0,00780
X_2=13	2197	0,13712
X_3=1	1	0,00006
X_4=7	343	0,02141
X_5=16	4096	0,25563
X_6=21	9261	0,57798
Toplam	16023	1,00000

Bu değerler, rulet tekerleğinin her çevrilişinde hangi olasılıkla hangi bireyin seçileceğini belirtir. Yani %58 olasılıkla 6. kromozom seçilecektir. Rulet tekerleği ve bireylerin tekerlek üzerindeki dağılımları Şekil 3.13'de gösterilmiştir.



Şekil 3. 13. Rulet Tekerlek Dağılımı

3) Toplumdaki birey sayısının sabit kaldığı varsayıldığından dolayı, rulet çarkının 6 defa çevrilmesiyle Tablo 3.8 de verilen kromozomlar seçilir. Bu noktada dikkat edilmesi gereken yer, böyle bir seçimde zayıf olan(uygunluk değeri küçük olan) kromozomlar rastgele rulet tekerleği seçimini geçememiştir. Çoğunlukla uygunluk değerleri büyük olan kromozomlardan seçilerek tablo 3'de gösterilen 6 üyeli yenilenmiş bir GA toplumu elde edilmiştir. Böyle bir rulet oyununda en fazla olasılıkla 6' ıncı kromozom yani 21 sayısının meydana gelmesi beklenir.

Tablo 3. 9. Kromozom Seçimi

Bireyler	**Kromozomlar**					
X_1=21	0	1	0	1	0	1
X_2=21	0	1	0	1	0	1
X_3=16	0	1	0	0	0	0
X_4=13	0	0	1	1	0	1
X_5=21	0	1	0	1	0	1
X_6=21	0	1	0	1	0	1

4) Tablo 3.9'da meydana gelen kromozomlar zaman içinde gelişmeleri için önce kendi aralarında çaprazlama oluşur. Örneğin başlangıçta çaprazlanacak kromozomlar ikilileri ve daha sonra da çaprazlamanın yapılacağı gen numaraları rastgele seçilir. Tablo 3.10 da gösterilmiştir.

Tablo 3. 10. Çiftleşme Grupları

Çaprazlanacak Kromozom Çiftleri	**Çaprazlama Noktaları**
X_6 ve X_2 kromozomları	3
X_5 ve X_4 kromozomları	2
X_1 ve X_3 kromozomları	5

Tablonun ilk sütununda eşleştirilen kromozomlar ikinci sütunda ise çaprazlama noktaları gösterilmiştir.

Çiftleşme Grubu -1

Ebeveyn 1

0 1 0 1 0 1

Ebeveyn 2

0 1 0 1 0 1

Nesil 1=010101(21) **Nesil** 2=010101(21)

Çiftleşme Grubu -2

Ebeveyn 1

0 1 0 1 0 1

Ebeveyn 2

0 0 1 1 0 1

Nesil 1=011101(29) **Nesil 2**=000101(5)

Çiftleşme Grubu -3

Ebeveyn 1

0 1 0 1 0 1

Ebeveyn 2

0 1 0 0 0 0

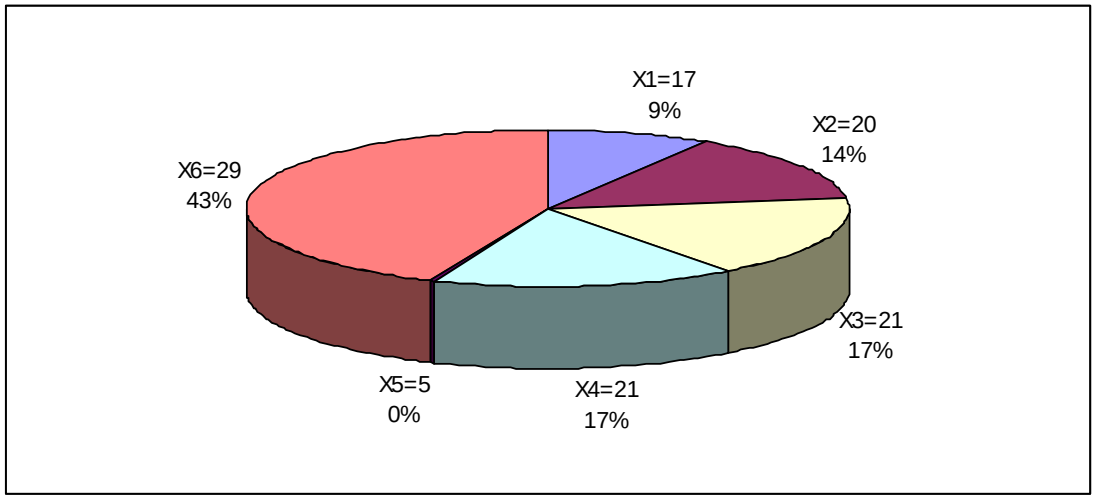
Nesil 1=010100(20) **Nesil 2**=010001(17)

Tüm çaprazlamalardan elde edilen sonuçlar Tablo 3.11'de verilmiştir.

Tablo 3. 11 Çaprazlanmış GA topluluğu

Bireyler	**Kromozomlar**					
X_1=17	0	1	0	0	0	1
X_2=20	0	1	0	1	0	0
X_3=21	0	1	0	1	0	1
X_4=21	0	1	0	1	0	1
X_5=5	0	0	0	1	0	1
X_6=29	0	1	1	1	0	1

Yeni oluşan popülasyon başlangıçtaki popülasyondan daha kuvvetlidir. Şekil 3.14' de yeni oluşan popülasyonun uygunlukları gösterilmektedir.



Şekil 3. 14 Yeni oluşan popülasyonun uygunluk değerleri

5) Son aşama olan mutasyon işlemi uygulanır. Mutasyona uğrayacak kromozomun geni rastgele olarak belirlenir. Bunun için 1 ile 6 arasında yapılan tam sayı seçimi 1 olsun. Mutasyona uğrayacak olan kromozom da 3.birey olarak rastgele seçilir. Buna göre mutasyona uğrayacak gen 3.kromozomun 1 numaralı genidir. Bu işlem tamamlandıktan sonra tablo 3.12' deki şekilde gerçekleşir.

Tablo 3. 12 Mutasyona geçirmiş GA popülasyonu

Bireyler	**Kromozomlar**					
X_1=17	0	1	0	0	0	1
X_2=20	0	1	0	1	0	0
X_3=53	1	1	0	1	0	1
X_4=21	0	1	0	1	0	1
X_5=5	0	0	0	1	0	1
X_6=29	0	1	1	1	0	1

Bu kromozom topluluğundan görüleceği üzere 3 nolu kromozomun değeri mutasyondan sonra 53'e ve fonksiyon değeri 53^3 'e ulaşmıştır. İlk baştaki en iyi kromozom 21 ve dolayısıyla fonksiyonda 21^3'tü.

3 temel operatörden oluşan genetik algoritma her aşamada yeni oluşan kuşağa uygulanarak bir sonraki kuşak elde edilecektir. Yukarıdaki örnekte tek bir iterasyon yapılmış ve başlangıç toplumundan bir sonraki kuşak oluşturulmuştur ancak genetik algoritmanın çalışmasının tam olarak gözlenebilmesi için tek bir iterasyon yeterli değildir. Yukarıdaki işlemlerde her şey çok fazla rasgele gibi görünse de, uygunluk değeri yüksek olan bireylerin seçilme ve çiftleşme olasılıkları yüksek olduğu için kuşaklar ilerledikçe toplumu oluşturan bireylerin uygunluk değerlerinin ortalamasının da arttığı gözlenecektir. Bunun için ise tek bir iterasyon yeterli değildir.

Şimdiye kadar olan adımlar tekrarlanırsa iyi kromozomun [1 1 1 1 1 1] (63) ve en iyi fonksiyon değerinin de 63^3 olacağı görülür.[31]

BÖLÜM 4. GENETİK ALGORİTMA İLE FIR (SONLU DÜRTÜ YANIT) FİLTRE TASARIMI

4.1. Giriş

Geleneksel optimizasyon algoritmaları çok boyutlu ve doğrusal olmayan problemlerin çözümünde veya optimizasyonunda istenilen sonuçları vermemektedir. Bu tür algoritmalar, karmaşık bir problemlerin çözümünde kullanılmamaktadır fakat geleneksel algoritmalar karmaşık problemlerin çözümünde kısmi olarak çalışırlar. Karmaşık problemlerin çözümünde genellikte rasgele optimizasyon algoritmalarına ihtiyaç duyulur. Bu algoritmalarında yönlendirilmiş ve yönlendirilmemiş türü vardır. Yönlendirilmemiş rasgele algoritma türünde optimum çözüm için geniş bir aralığa bakılacağı için çözümü bulma süresi artmaktadır.

Genetik algoritma, yönlendirilmiş rasgele algoritma türlerindendir. Genetik algoritma başlangıçta arama aralığından elde edilen başlangıç popülasyonunu kullanır. Popülasyon doğal seçilim ve tekrar üretim işlemleri uygulandıktan sonra popülasyonun uygunluğa olan yakınlığı artmaktadır. Popülasyonun daha da iyileştirilmesi için genetik algoritma operatörleri uygulanır ve elde edilen popülasyon çözüme yakın olan popülasyondur.

Genetik algoritmanın temel avantajı olarak optimize etmeye çalıştığı problemin çözümünde karmaşık matematiksel işlemlere ve ek bilgiye ihtiyaç duymamasıdır. Genetik algoritmanın temelinde diğer algoritma çeşitlerinden farklı olarak ve algoritmanın etkisini artıran temel 4 parametreye sahiptir. Bu parametreler gösterim, başlangıç popülasyonu, uygunluk fonksiyonu, genetik operatörlerdir.

Genetik algoritmalar doğrusal olmayan, karmaşık ve ayrık(sür”ekli olmayan) problemlerin çözümünde diğer optimizasyon tekniklerine göre ideal’e yakın

çözümler üreten güçlü bir optimizasyon algoritmasıdır. Genetik algoritmanın(GA) diğer arama yöntemlerine göre avantajları aşağıda verilmiştir.

5) GA işlemleri türev ve integral gibi hesaplamalar gerektirmez ve tüm hesaplamalar aritmetik işlemlere dayanır. Karar uzayının taranması sırasında önemli etkiler sadece hedef fonksiyonu ve ona bağlı olarak ortaya çıkan uygunluk dereceleridir.
6) GA karar uzayında aynı anda ve paralel olarak birçok noktada değerlendirme yaparken, bu durum klasik yöntemlerde tek nokta ve yön olarak karşımıza çıkar.
7) GA işlemleri sırasında belirgin(deterministik) değil ihtimali ve stokastik geçişler kullanılır.
8) GA'da değişkenler gerçek ondalık değerleri olabilecek çok fazla sayıda çözüm getirir.

4.2. Genetik Algoritma ile Sonlu Dürtü Yanıtı(FIR) Filtre Katsayılarının Hesaplanması

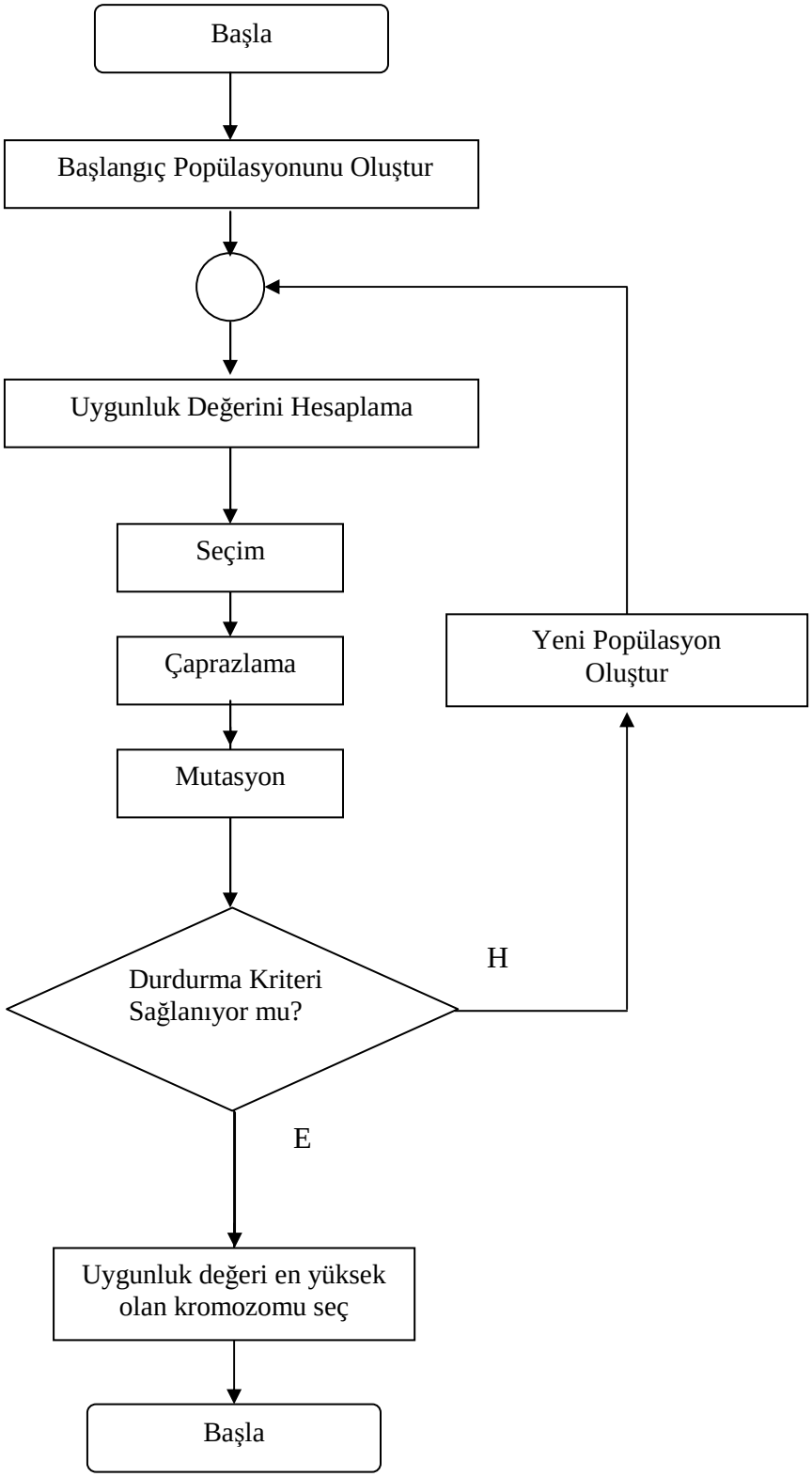
Sayısal filtre tasarımında istenen filtrenin özelliklerine göre, filtre hakkında bilgi veren transfer fonksiyonda bulunan katsayıların hesaplanmasıdır. Filtrelerin çeşitleri yani alçak geçiren, yüksek geçiren, band geçiren ve band durduran filtreye göre transfer fonksiyonunda kullanılan katsayılarda değişmektedir. Bu çalışmada sonlu dürtü yanıtı filtrenin çeşitlerinin katsayıları genetik algoritma kullanılarak elde edilmiştir.

4.2.1. Genetik algoritmaların çalışma prensibi

Genetik algoritmalar doğal seçim ilkesine dayanan bir sayısal optimizasyon yöntemidir. Genetik algoritma, çözüm dizilerinden oluşan bir başlangıç nesliyle, çaprazlama ve mutasyon gibi doğal seçim operatörlerini kullanmaktadır.

Genetik algoritmalar oldukça genel prensiplerle Şekil 4.1'de akış şemasında görüldüğü gibi çalışmaktadır. Öncelikle ele alınan problem için bir rasgele *n* kromozomlu popülasyon oluşturulur. Daha sonra popülasyondaki her bir kromozom

için *f(x)* uygunluk fonksiyonu hesaplanır. Yeni bir popülasyon oluşuncaya kadar aşağıdaki adımlar tekrar edilir:



Şekil 4. 1. Genetik algoritmanın akış şeması

Akış şemasının adım adım açıklanması:

1. [Başlangıç]: n kromozom oluşan rasgele bir popülasyon oluşturulur (problemin olası çözümleri)
2. [Uygunluk]: f(x) fonksiyonunda popülasyonun her bir kromozomu(x) değerlendirilir.
3. [Yeni Popülasyon]: Yeni popülasyon tamamlana kadar aşağıdaki adımların tekrar edilmesiyle yeni popülasyon oluşturulur.

a. [Seçilim(Seleksiyon)]: Popülasyondan iki ebeveyn kromozom seçilir.(daha uygun olanın seçilme sansı daha fazladır)

b. [Çaprazlama]: Çaprazlama olasılığıyla yeni nesil oluşturmak için eşleştirilir. Eğer hiç çaprazlama olmazsa çocuklar ebeveynlerinin aynıları olur.

c. [Mutasyon]: Mutasyon olasılığı ile her yerdeki yeni nesil'i mutasyona uğratır.

4. [Durdurma Kriteri]: Eğer son durum yeterliyse en iyi kromozom seçilir veya popülasyona yeni bireyler eklenmesi için döngünün başına döner.
5. [Döngü]: 2. adıma geri dönülür.

4.2.2. Genetik algoritmaların FIR filtre katsayılarına uygulanması

Sonlu dürtü yanıt katsayılarının hesaplanması için kullanılan fonksiyonlar filtre çeşitlerine göre değişmektedir. Bu çalışma da filtre türlerinden alçak geçiren, yüksek geçiren, band geçiren ve band durduran filtre kullanılmaktadır.

Genetik Algoritma Filtre Katsayılarına Uygulanması

- Başlangıç Popülasyonu: Uygunluk fonksiyonunda kullanılacak bireylerin elde edilmesi için başlangıçta rasgele olarak 0 ile 1 arasında üretilen bir topluluktan oluşur.

- Uygunluk Fonksiyonu: Filtrelerdeki fonksiyonlar şartları sağlayacak şekilde uygunluk fonksiyonuna çevrilmiştir. Uygunluk fonksiyonu optimizasyon yapmak istediğimiz fonksiyondur yani amaç fonksiyonumuzdur. Örnek alçak

geçiren filtrenin amaç fonksiyonları denklem 4.1'de gösterilmiştir. Şekil 4.2'de gösterilen 2 ayrı fonksiyon birincisi alçak geçiren filtrenin orta değerini bulan, diğeri ise diğer katsayıları bulan uygunluk fonksiyonlarıdır.

Alçak geçiren filtrenin orta katsayının uygunluk fonksiyonu:

```
function y=h0_alcak(i)
  global wc1
    y=((wc1*i)/(pi*i));
end
```

Alçak geçiren filtrenin diğer katsayıların uygunluk fonksiyonu:

```
function y=ga_alcak(i)
global wc1
      y=(sin(wc1*i)/(pi*i));
end
```

Şekil 4. 2. Alçak geçiren FIR filtrenin katsayıların hesaplanması için kullanılan fonksiyonlar

Şekil 4.2 de gösterilen orta katsayı ve diğer katsayı değerlerinin uygunluk fonksiyonlarından popülasyon sayısı kadar üretilir. Yapılandırılan ayarlamalara göre değer üretir. Bu değerler arasında çiftleştirir. Çaprazlama ve mutasyon yoluyla da katsayı değerlerinin değerlerini çeşitleştirir.

Örnek olarak orta katsayı değeri için genetik algoritmayı çalıştırdığımızda;

Popülasyon sayısı: 20 olarak seçilmiştir. Bu kısım her nesildeki popülasyonun genişliğini belirtir. Popülasyon çeşitliliği artırmak genetik algoritmanın daha çok nokta araştırmasının ve daha iyi bir sonuç almasını kolaylaştırır. Bununla birlikte popülasyon genişliği ne kadar genişse genetik algoritmanın her nesilde işlemesi daha uzundur.

— Seçim**:** Seçilim fonksiyonu uygunluk sıralama fonksiyonundan sıralanmış değerlere bağlı kalarak yeni nesil için ebeveynler(katsayılar) seçilir. Üretilecek katsayılar daha iyi uygunluk değerine sahip bireyler tarafından

gelecek bireyler (katsayılar) oluşturulur. Programda seçilim fonksiyonu olarak tek noktalı, stokastik, rank, rulet ve turnuva sıralaması kullanılmıştır.

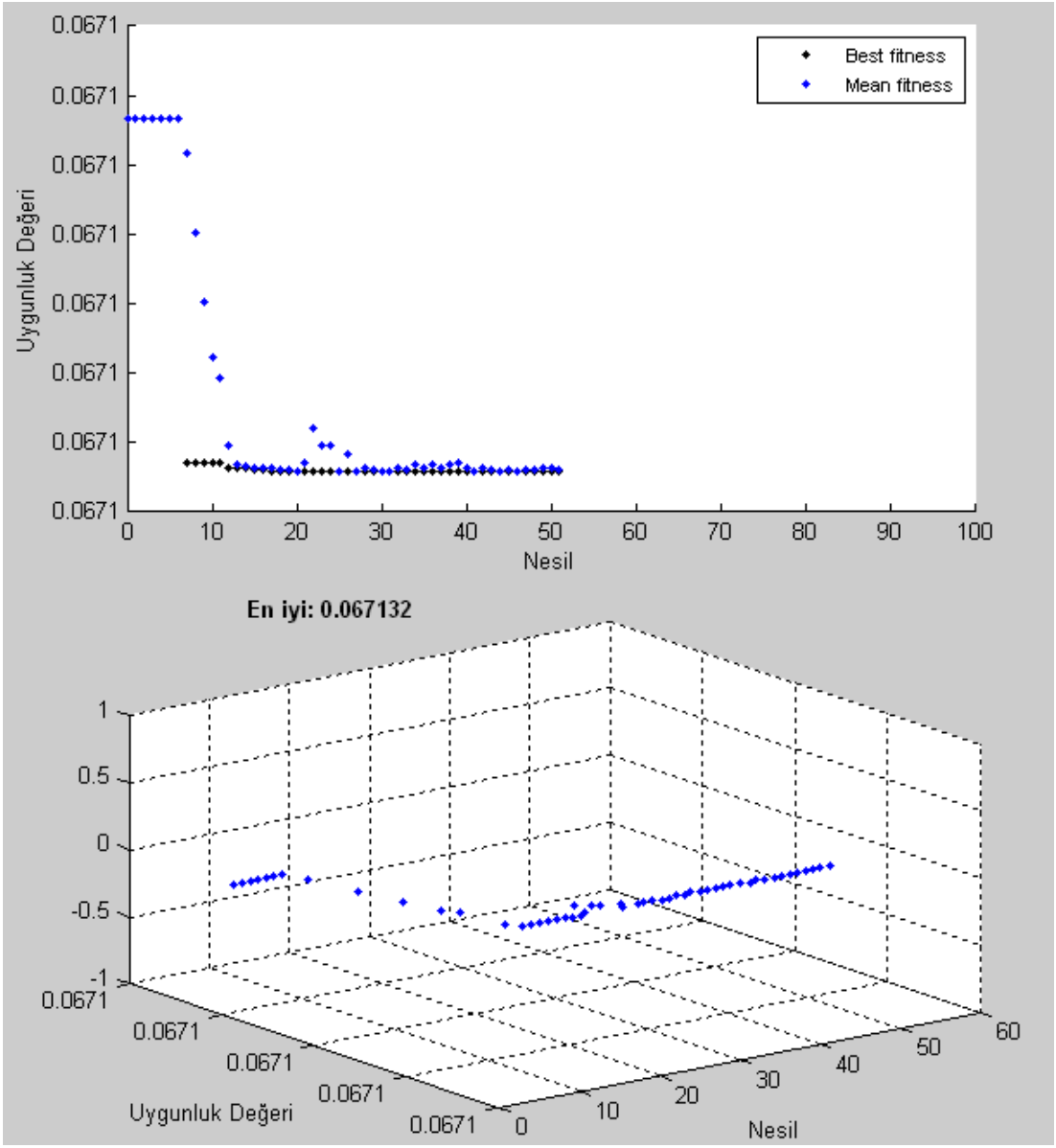
Bu örnekte seçim “Stochastic uniform” kullanılmıştır. Bu sıralamanın özelliği ise “Stochastic uniform selection” gibi sıralanmış değerlerin bölümsel parçalarını kullanarak ek ebeveynleri seçer. Fonksiyon bireylerin sıralanmış değerlerinin bölümlü parçalarına orantılı olan uzunluklu bölümler içeren çizgi oluşturan ve ebeveynleri seçmek için eşit adımlarlar hat boyunca hareket eder.

— Mutasyon ve çaprazlama: Nesiller sıralandıktan sonra, en iyi uygunluk değerine sahip olan bireyler(katsayılar) ile çaprazlama ve mutasyon uygulanarak yeni bireyler elde edilir. Çaprazlama da mevcut nesilden bir çift bire seçilir ve yeni bireyler oluşturmak için bu çiftler birleştirilir. Mutasyon da ise mevcut nesildeki bir tek bireye uygulanan rasgele değişimlerdir. İşlemlerin ikisi de genetik algoritma için önemlidir. Çaprazlama farklı bireylerden en iyi genleri çekmek için algoritmaya yardımcı olur ve daha üstün çocuklar oluşturmak için yeniden birleşirler. Mutasyon popülasyona çeşitlilik katar ve daha iyi uygunluk değeriyle algoritmanın yeni bireyler oluşturma olasılığı artar.

— Gelecek nesil oluşturulurken kaç nesil değiştirerek en iyinin elde edileceğini nesil sayısından elde edebiliriz.

Şekil 4.3’de alçak geçiren FIR filtrenin orta katsayı değeri için en iyi uygunluk değeri hesaplanmıştır. Bu örnekte Şekil 4.2’de gösterilen fonksiyonlar kullanılmıştır ve kesim frekans değeri olarak da 0.2125 seçilmiştir.

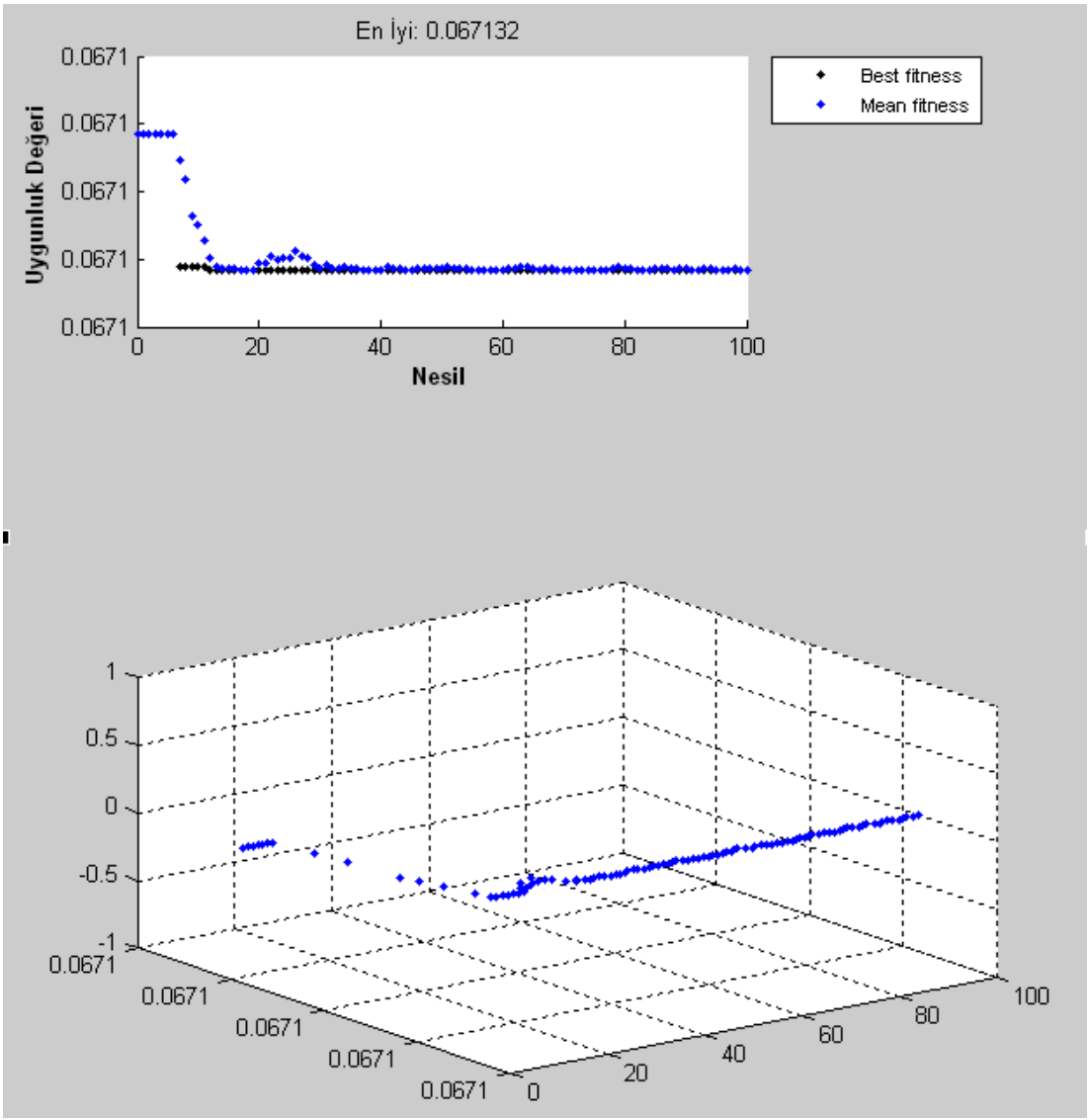
MATLAB programında orta değer için yapılan genetik uygulamasında belirtilen popülasyon büyülüğü içerisinde 0.0671, 0.0678, 0.0672, 0.0673, 0.0674, 0.06733, 0.06722, 0.06755, 0.0676, 0.0677… gibi 20 tane değer elde edilmiştir. Seçim, çaprazlama ve mutasyon işlemlerinden geçtikten sonra en iyi uygunluğa sahip olan 0.0671 seçilmiştir, Şekil 4.3 de açık bir şekilde görülmektedir.



Şekil 4. 3. En iyi neslin gerçekleştirilmesi

Şekil 4.3'de algoritma başlangıç popülasyonda bireylerden gen seçer ve onları tekrar birleştirir. Algoritma en iyi uygunluk çizimin seviyeleştiği nesil sayısı 8 de bu genleri kullanarak en iyi bireyi oluşturur. Bundan sonra nesil için seçilen en iyi bireyin yeni kopyaları oluşturur. Nesil sayısı 12 olduğunda popülasyondaki bütün bireyler aynıdır ve bu nesil'i en iyi birey olarak adlandırabilir. Bu durumda bireyler arası ortalama uzaklık 0'dır. Algoritma nesil 8 den sonra en iyi uygunluk değerini geliştiremez. Bu durum 50 nesil için sürer. Bu şekildeki en iyi uygunluk değeri sabit kaldığı için nesil sayısı artırılsa bile en iyi uygunluk değeri değişmez. Şekil 4.4'de nesil sayısı arttırılırsa bile en iyi değer bulunduktan sonra

değişmediği gösterilmiştir. Durdurma kriteri de neslin durdurulmasıyla gerçekleştirilmiştir.



Şekil 4. 4. Mevcut neslin artırılması

BÖLÜM 5. ARAYÜZ TASARIMI

5.1. Giriş

Günümüzde mühendislik sistemleri ve problemlerinin çözümünde bilgisayar kullanımı kaçınılmaz hale gelmiştir. Problemlerin çözümünde zaman kaybını azaltmak ve verimi artırmak için farklı yöntemler geliştirilmiştir. Tekrarlı hesaplamalar için en uygun çözüm bilgisayar programları ile gerçekleştirilmesidir. Bilgisayar destekli eğitim, mühendislik ve teknik eğitimde teorik bilgilerin laboratuar ortamına benzetilerek geliştirilmesinden dolayı fiziki koşulların eksikliği giderilir. Laboratuar oluşturulması maddi sıkıntıdan dolayı gerçekleştirilememesi bilgisayar ortamında eğitim amaçlı sanal laboratuarların geliştirilmesine neden olmuştur. Bu laboratuarlar eğitimde esnekliği, etkileşimi ve tekrar tekrar gerçekleştirilmesini sağlar[36].

Bilgisayar destekli eğitim ortamların sağlamış olduğu avantajları aşağıdaki gibi sıralanabilir.

- Çağın gereklerine ve teknolojik gelişime uygun bir eğitim ortamı oluşturulabilir.
- Zaman ve mekân sorunları ortadan kaldırılarak, öğrenciler eğitim kurumuna değil eğitim öğrencilere götürülmüştür. Bu sayede eğitimi daha geniş kitlelere yaymak mümkündür.
- Öğrenciler, sadece bir tuşa tıklamak ile düşünmeye, araştırmaya ve akılcı fikirleri sınamaya yöneltilir.
- İşlemlere uygulamalar ile öğrenme süreci daha etkili, verimli ve çekici bir deneyim niteliğine kavuşturulabilir
- İşlemlere defalarca tıklanarak, uygulamalar tekrar tekrar gerçekleştirilebilir.

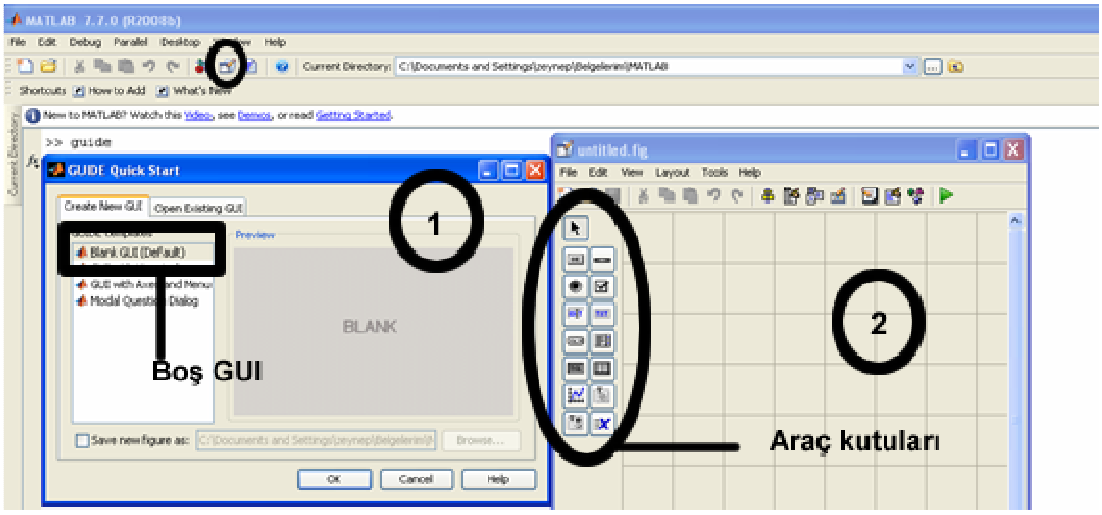
Bu bölümde genetik algoritma kullanılarak elde edilen FIR filtre katsayılarının hesaplanmasında ve elde edilen değerler ile filtrenin cevaplarının grafiksel olarak elde edilmesine yönelik iki farklı arayüz tasarlanmıştır. Bu arayüzlerden biri MATLAB GUI ile diğeri ise ASP.NET ile hazırlanmıştır. İki farklı arayüzden elde edilen sonuçlar karşılaştırılmıştır.

5.2. MATLAB GUI ile Hazırlanan Arayüz

1985 yılında C.B Moler tarafından geliştirilen MATLAB(MATrix LABoratory) adlı yazılım geliştirme aracında teknik hesaplamalar ve matematiksel problemlerin çözümünde ve analizinde kullanılır. Matlab programı matris işlemlerinin gerçekleştirilmesinde özellikle tercih edilir. MATLAB mühendislik alanı, sistem analizi, görüntü işleme(image processing), yapay sinir ağları(artificial neural networks), sayısal işaret işleme(signal processing), optimizasyon (optimization), veri elde etme(data acquisation), veritabanı(database), süzgeç tasarımı (filter design), bulanık mantık (fuzzy logic), sistem tanımlama (system identification), dalgacıklar (wavelets) gibi araçları geliştirme ortamı sunar.

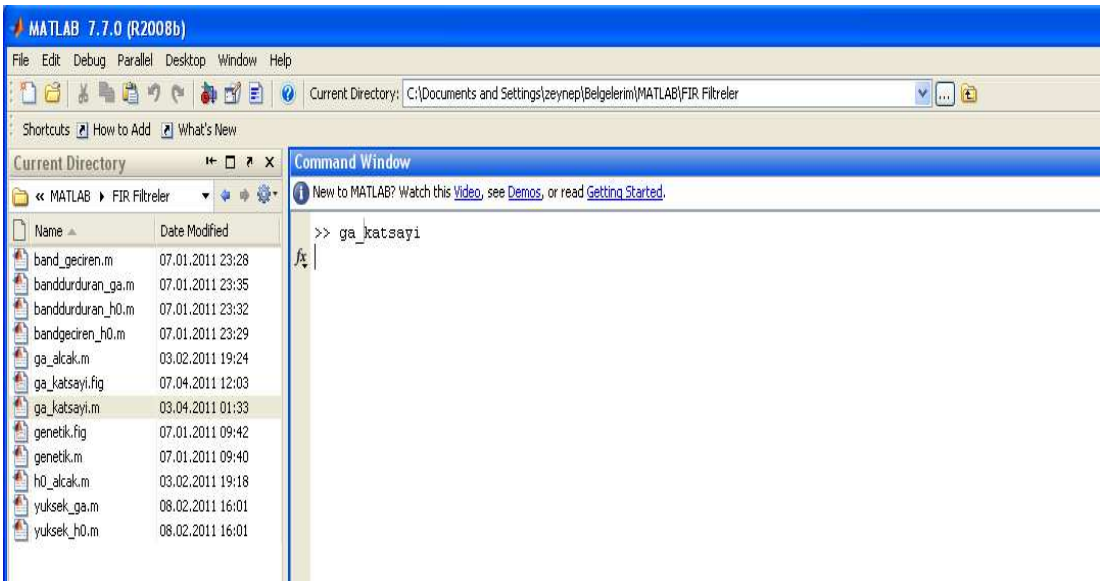
MATLAB'da yazılan programlar, MATLAB'ın kendi dilinde yazılabileceği gibi DLL ve EXE olarak da oluşturulabilir ve C/C++ kodlarına çevrilebilir. MATLAB da programlar komut satırında yazılabilir ve görsel olarak tasarım yapmayı sağlayan MATLAB geliştirme aracı kullanılarak gerçekleştirilebilir. MATLAB GUI'de bulunan textbox, button, checkbox ve combobox gibi görsel programlama araçları ile fonksiyonlar ve komutlar uyumlu bir şekilde çalışabilir.

Bu tez çalışmasında da MATLAB GUI aracının özellikleri kullanılarak sonlu dürtü yanıtı(FIR) filtrenin çeşitleri geliştirilmiştir. MATLAB GUI aracına ulaşmak için çalışma ekranına "guide" yazılır veya Şekil 5.1'de yuvarlak içine alınmış ikona tıklatılarak çalıştırılabilir. Ekrana gelen pencereden (Şekil 5.1'deki 1 nolu pencere), yeni bir çalışma için "BLANK GUI(Default)" seçeneği seçilerek ekranaboş bir tasarım penceresi (Şekil 5.1'deki 2 nolu pencere) açılır. Bu pencerenin sol tarafındaki araçlar kullanılarak istenen arayüz tasarımı gerçekleştirilebilir. GUI arayüzüyle gerçekleştirilen tasarımların "figure" olarak kaydedilebilir. [35]



Şekil 5. 1. MATLAB GUI'nin başlatılması

Bu tez çalışmasında yapılan GUI tasarımı "ga_katsayi" olarak adlandırılmıştır. Bu arayüz tasarlanırken "Current Directory" de görülen m dosyalarıyla gerçekleştirilmiştir. Arayüzü çalıştırmak için komut satırına "ga_katsayi " yazılması yeterli olacaktır. Burada dikkat edilmesi gereken "current directory" bölümüne çalışma dosyasının yönlendirilmesi gereklidir.(Şekil 5.2)

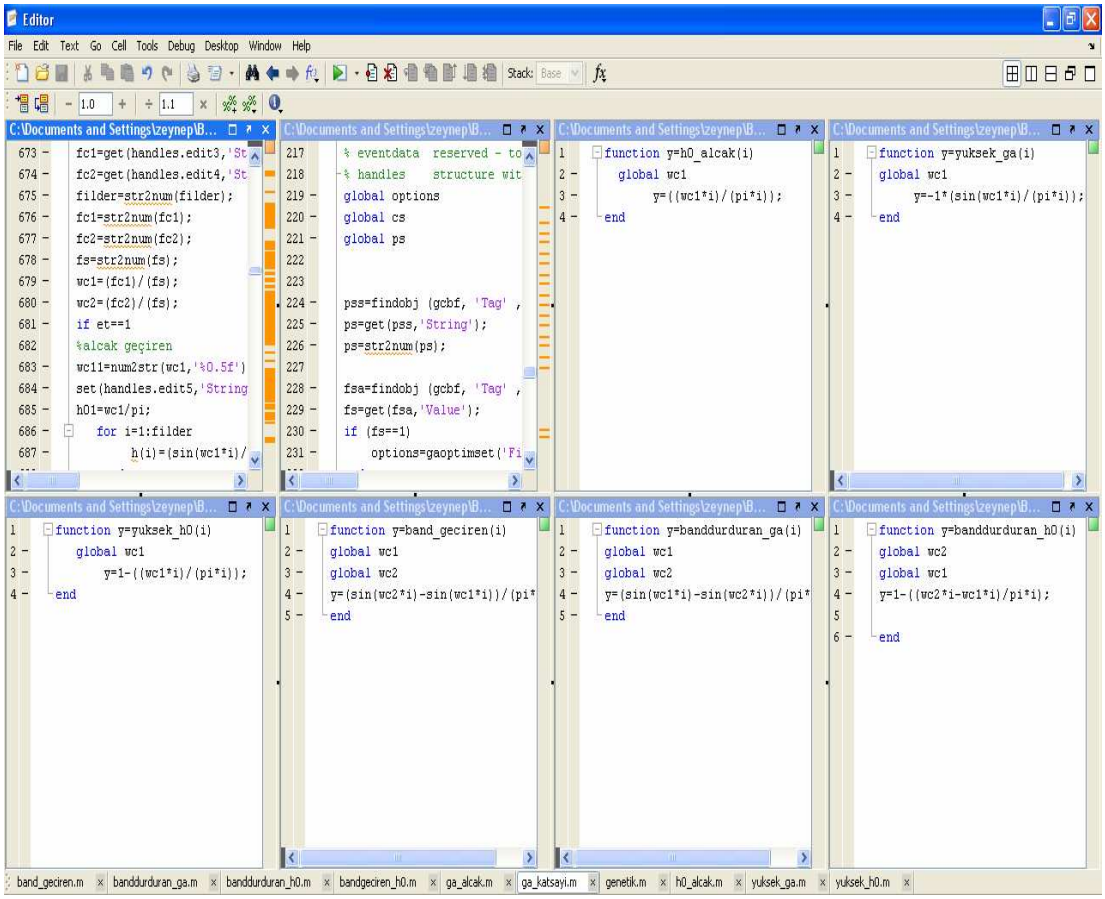


Şekil 5. 2. Arayüzün içerdiği dosyalar ve arayüzün çalıştırılması

Şekil 5.2'de, tasarlanan arayüzde kullanılan dosyalar görülmektedir. Arayüzün çağrılması için komut satırına "ga_katsayi" yazılması gereklidir."ga_katsayi"

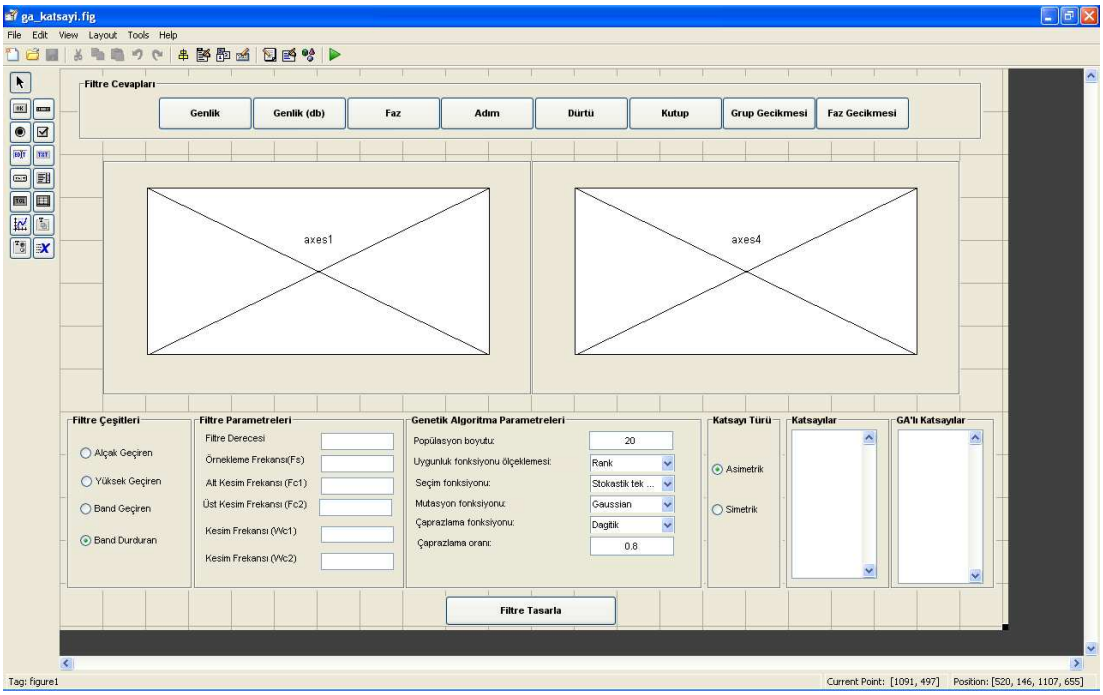
dosyasında bütün komutlar içerisine gömülmüştür. Aynı zamanda genetik algoritma ile gerçekleştirilen fonksiyonlar bu kısımda çağrılarak gerçekleştirilebilir.

“ga_katsayi .m” dosyasına gelen harici dosyalar da genetik algoritma kullanılarak elde edilmiş fonksiyonlar bulunur. Filtre çeşitleri ayrı .m dosyalarına fonksiyon olarak kaydedilmiştir. Burada dikkat edilmesi gereken filtrelerin katsayı değerleri elde edilirken orta katsayı değeri ve diğer katsayıları gerçekleştiren fonksiyonlar birbirinden farklıdır. Örneğin alçak geçiren filtre için orta katsayı değerini hesaplamada kullanılan fonksiyon “h0_alcak.m” diğer katsayıları hesaplamada kullanılan fonksiyon ise “ga_alcak.m” dır.”ga_katsayi.m ” dosyasına gömülen komutlarda filtre çeşitlerinin katsayıları elde edilir, bu katsayılar kullanılarak elde edilecek frekans cevabı, adım cevabı, dürtü cevabı, kök eğrileri ve grup gecikmesi ve faz gecikmesini veren komutlar bulunmaktadır. Şekil 5.3’de arayüzde temel olarak kullanılan .m dosyaları bulunmaktadır.

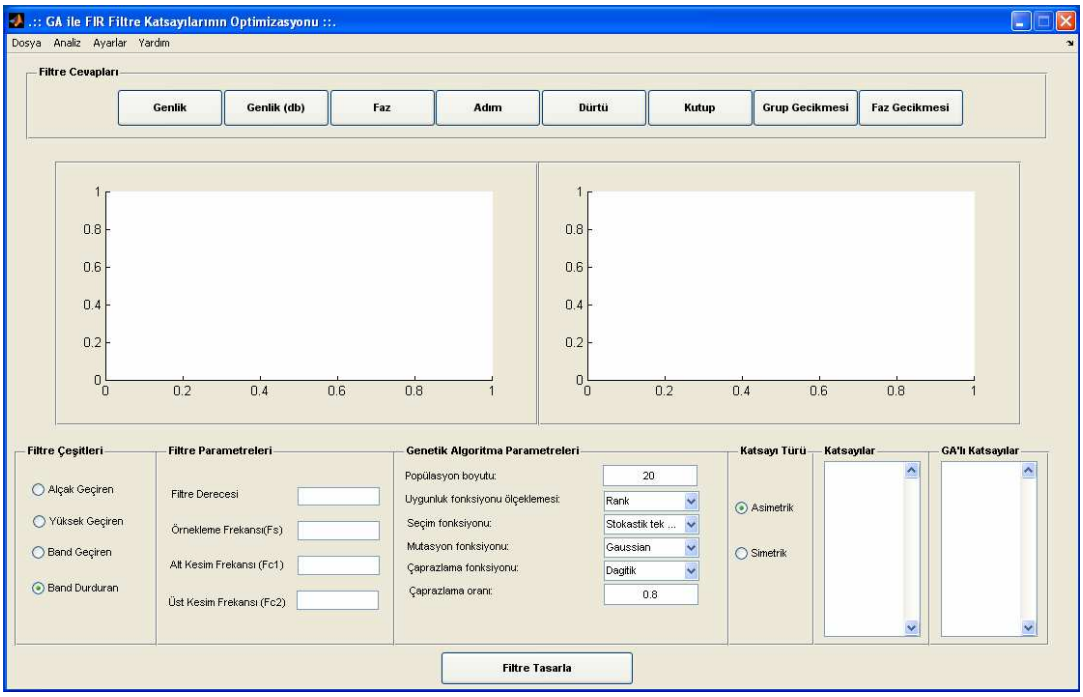


Şekil 5. 3. Arayüzde kullanılan .m dosyaları ve fonksiyonları

Şekil 5.4’ de MATLAB GUI ile tasarlanan arayüzün ilk hali görülmektedir. Şekil 5.4’ deki tasarımı çalıştırmak için Debug menüsünden run sekmesi seçilir veya Şekil 5.4’te yuvarlak içerisine alınan ikon seçilerek çalıştırılabilir. Butonlar kullanılarak elde edilen filtre cevapları aynı zamanda menüler kullanılarak da gerçekleştirilebilir. Filtre cevapları adı altında bulunan cevaplar da üzerine tıklandığı zaman aynı komutları çalıştırır.



Şekil 5. 4. MATLAB GUI ile FIR Filtreler için tasarlanan arayüz



Şekil 5. 5 Arayüzün çalıştırılmış hali

Arayüzün tasarlanma amacı sayısal filtre çeşitlerinden sonlu dürtü cevabı(FIR) filtrenin çeşitlerinin katsayılarını hesaplamak ve farklı filtre cevaplarını elde etmektir. Ayrıca ideal olan FIR filtre katsayılarını genetik algoritma da kullanılarak elde edilmiştir. Elde edilen katsayılar karşılaştırılmıştır.

Şekil 5.5' deki şekil de görüldüğü üzere arayüzde farklı kısımlar bulunmaktadır. "Filtre Çeşitleri " kısmında FIR (sonlu dürtü cevabı) filtrenin çeşitlerinin seçilmesi kısmıdır. Filtre çeşitlerinden alçak geçiren, yüksek geçiren, band geçiren ve band durduran filtreler tasarlanabilir. Yapılan seçime göre programda ilgili olan kod parçacıkları girilen parametrelere göre çalıştırılır.

"Filtre Parametreleri" kısmından filtrenin derecesi, örnekleme frekansı, filtre çeşidine göre alçak ve yüksek geçiren filtrelerde alt kesim frekansı değer girilir, band geçiren ve band durduran filtrede ise alt ve üst kesim frekansı girilir.

"Katsayı Türü" kısmında filtre çeşitleri 4 çeşit olduğu için bu kısımda asimetrik katsayılar veya simetrik katsayılar elde edilmesi için seçim gerçekleştirilir.

“Genetik Algoritma Parametreleri” kısmından genetik algoritma ile ilgili parametreler değiştirilebilir. Popülasyonun boyutu ayarlanabilir, varsayılan değer de 20’dir. Uygunluk fonksiyonun ölçeklenmesi kısmında uygunluk fonksiyonları elde edildikten sonra sıralanma şeklini gösterir. Açılır menüden Rank, Oransal, Tepe, Doğrusal değişim seçeneklerinden biri seçilebilir. Varsayılan değer olarak “Rank” seçilmiştir. Seçim fonksiyonu sekmesinden de Seçilim fonksiyonu uygunluk sıralama fonksiyonundan sıralanmış değerlere bağlı kalarak yeni nesil için ebeynler(katsayılar) seçilir. Burada Stokastik Tek biçimli, Artan, Tek biçimli, Rulet ve Turnuva gibi seçim fonksiyonları seçilebilir. Mutasyon fonksiyonu sekmesinden seçilen katsayıların çeşitliliğin artması için uygulanan fonksiyon türüdür. Mutasyon fonksiyonu olarak Gaussian, Tek biçimli ve Adaptif seçilebilir. Çaprazlama fonksiyonu sekmesine ise katsayılara uygulanacak çaprazlama fonksiyonu seçilir. Fonksiyonlar, Dağıtık, Tek Noktalı, İki noktalı, Ara, Heuristik ve Aritmetik seçilebilir. Çaprazlama oranının değeri de girilebilir. Varsayılan çaprazlama oranı miktarı 0.8 olarak ayarlanmıştır.

“Katsayılar” kısmından ilk önce filtre çeşidi, katsayı türü ve filtre parametreleri girildikten sonra ilgili kod parçacığı ile filtrenin derecesine göre üretilen katsayıların gösterildiği bölümdür. “GA’ lı Katsayılar” kısmı genetik algoritma parametreleri de kullanılarak elde edilen katsayıların bulunduğu bölümdür.

“Filtre Cevapları” kısmında katsayılar kullanılarak elde edilen genlik cevabı, db cinsinden genlik cevabı, faz cevabı, adım cevabı, dürtü cevabı, kutuplar, grup gecikmesi, faz gecikmesi gibi cevaplar elde edilebilir. Hesaplanan ve genetik algoritma kullanılarak elde edilen çizimler ayrı çizim alanlarında gösterilir.

— Genlik Cevabı: Bir transfer fonksiyonunun frekans yanıtını veren grafiktir. Frekans cevabını elde etmek için freqz komutu kullanılmıştır. Hesaplanacak frekansları içeren bir ω vektörü tanımlanır. ω vektöründe bizim belirlemiş olduğumuz sınırlar bulunmaktadır. Genel olarak örneklerde ω=0:0.01:0.5 kullanılmıştır. ω’deki her bir frekans için frekans yanıtı değerini içeren karmaşık bir H vektöründe bulunur. Genlik cevabı db cinsinden kullanıldığında mag2db komutu seçilmiştir.

— Kutuplar: Sistem fonksiyonunun kutuplarına bakılarak sistemin kararlılığı hakkında sonuca ulaşılabilir. Sistemin kutupları, sanal eksenin üzerinde veya solunda bulunuyor ise sistem kararlı, aksi halde kararsızdır. Bu işlem MATLAB'da yer alan "zplane" komutuyla gerçekleştirilmiştir. Bu komutun kullanımında sisteme ait transfer fonksiyonu pay ve payda değerleri bir polinom formu oluşturacak şekilde ifade edilerek MATLAB'da vektörel formda tanımlaması yapılarak kullanılır. "zplane" komutunun kullanımı sonucunda elde edilen grafiksel sonuçta kutuplara ait kök değerleri "x" karakteri ile s-düzlemi üzerinde gösterilir iken sıfırlara ait kök değerleri "o" karakteri ile gösterilir. Sisteme ait kök değerlerini gösteren grafiksel sonuca ait eksen açıklamaları ve grafiği tanımlayan başlık İngilizce olarak oluşur. İstenildiğinde bu açıklamalar mevcut programa "xlabel", "ylabel" ve "title" komutları ile Türkçe karşılıkları yazılarak ekleme yapıldığında değiştirilebilir.

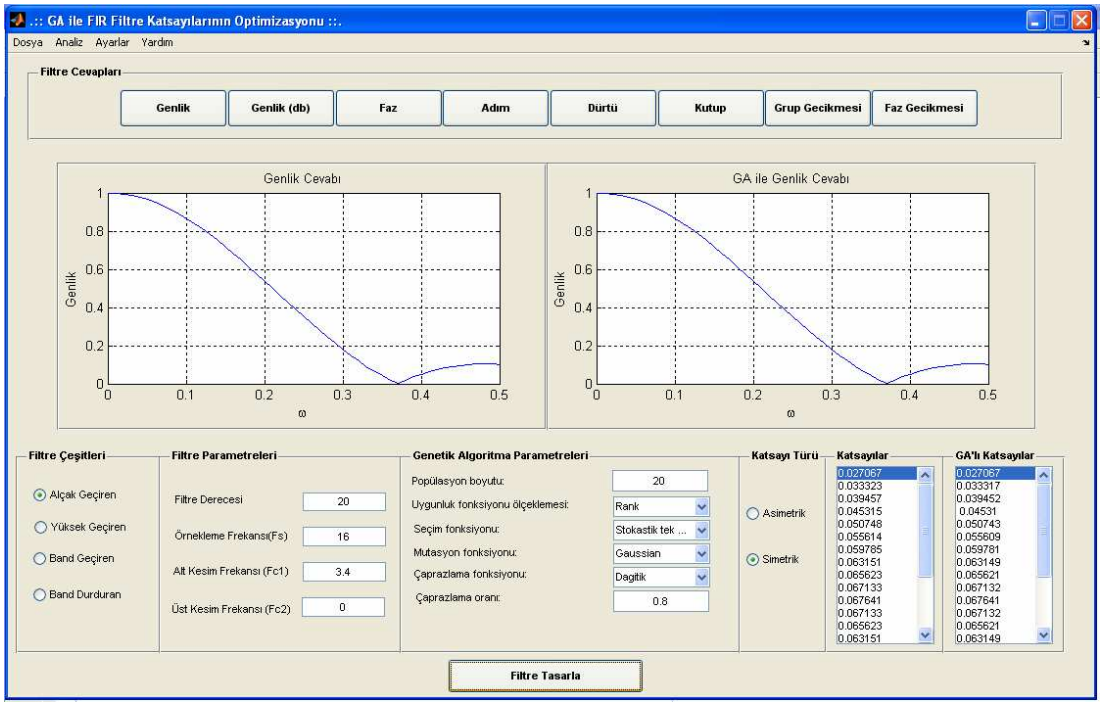
— Adım cevabı: Sistemin girişine birim basamak işareti uygulandığında çıkıştaki işareti (sistem cevabını) veren grafiktir. Gerçekleştirilen bu çalışmada adım cevabının karşılığı standart "stepz" komutunun kullanım yapısı tercih edilerek çizilmiştir. "Stepz" komutunun tercih edilmesinin nedeni ayrık zamanlı sistemler kullanıldığı için "step" komutu kullanılamamıştır yerine "stepz" komutu tercih edilmiştir. Bu yapı köklerin yer eğrisinde olduğu gibi sisteme ait transfer fonksiyonunun pay ve payda değerlerinin vektörel formda tanımlanmasıyla sağlanır. Grafiksel sonuca ait zaman dilimi ve değişim aralığı "stepz" komutu tarafından otomatik olarak belirlenir.

— Dürtü cevabı: Sistemin bir dürtü(birim ani) cevabını veren grafiktir. Gerçekleştirilen bu çalışmada ayrık zamanlı noktaların dürtü cevabını elde etmek için "impz" komutu kullanılmıştır. Bu yapı köklerin yer eğrisinde olduğu gibi sisteme ait transfer fonksiyonunun pay ve payda değerlerinin vektörel formda tanımlanmasıyla sağlanır. Grafiksel sonuca ait zaman dilimi ve değişim aralığı "impz" komutu tarafından otomatik olarak belirlenir.

— Faz Cevabı: Sonlu ve sonsuz dürtü cevaplı filtrelerin faz yanıtlarını elde ederken "angle" komutundan yararlanılmıştır. "Angle" komutunu kullanırken freqz komutundan elde edilen frekans yanıtı kullanılarak gerçekleştirilir.

Sistemin kararlılığı grafik üzerinden de elde edilebilir. Eğri FIR (sonlu dürtü cevaplı) filtre için doğrusala yakın ancak doğrusal değildir.

Eşitlik 2.13'deki denklem ile ifade edilen simetrik olan alçak geçiren FIR filtrenin katsayılarını elde etmek için kullanılır. Bu denklemleri kullanarak tasarlanan arayüz aracılığıyla filtre tipi alçak geçiren seçilir, katsayı türü olarak simetrik katsayılar ve filtre parametreleri olarak filtrenin derecesi 20, örnekleme frekansı 16 kHz, alt kesim frekansı 3.4 kHz olarak ayarlanmıştır. Genetik Algoritma parametreleri varsayılan olarak kullanılmıştır. Parametre girişleri yapıldıktan sonra "Filtre Tasarla" butonuna basılarak Şekil 5.6'daki şekil elde edilir.



Şekil 5. 6. Simetrik alçak geçiren filtrenin katsayıları optimizasyonu

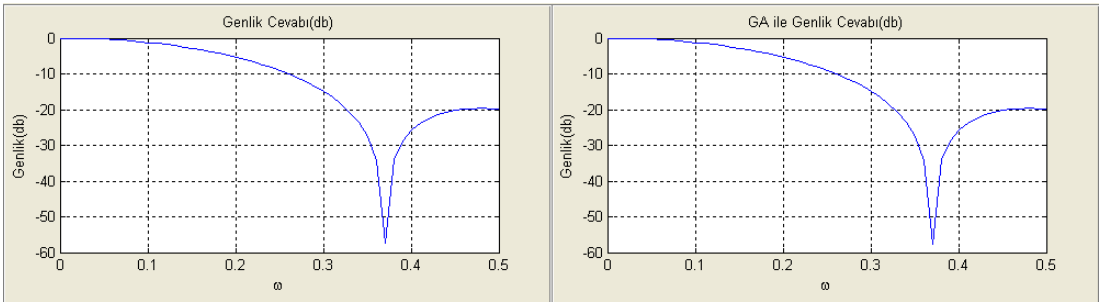
```
if kt==2
    if et==1
        %simetrik ideal alçak geçiren
     for i=1:yar1
        ht(i)=(sin(wc1*(i-yar1))/(pi*(i-yar1)));
     end
     ht(yar1)=wc1/pi;
     for ii=1:yar
        htt(ii)=(sin(wc1*ii)/(pi*ii));
     end
    th=[ht htt];
liste1=findobj(gcbf,'Tag','gliste1');
set(liste1,'string',num2str(th'));
%simetrik genetik alçak geçiren
 for k=1:yar1
[x,fval,reason,output,population]=ga(@simetrik_ga_alcak1,1,[],[]
,[1],[k],[1],[yar1],[],options);
    qa(k)=fval;
 end
[x,fval,reason,output,population]=ga(@simetrik_h0_alcak,1,[],[],
[1],[1],[1],[filder],[],options);
    gss0=num2str(fval,'%0.10f');
    gsss0=str2num(gss0);
 for q=1:yar
[x,fval,reason,output,population]=ga(@simetrik_ga_alcak2,1,[],[]
,[1],[q],[1],[yar],[],options);
    qa2(q)=fval;
 end
sgal1=[qa qa2];
liste2=findobj(gcbf,'Tag','gliste2');
set(liste2,'string',num2str(sgal1'));
%simetrik alçak geçiren filtrenin genlik cevabı
th1=freqz(th,1,w);
axes(handles.axes1);
th1=abs(th1);
th1=th1./max(th1);
plot(w,th1);
title('Genlik Cevabı');
xlabel('w');
ylabel('Genlik');
%simetrik genetik alçak geçiren filtrenin genlik cevabı
sgal11=freqz(sgal1,1,w);
axes(handles.axes4);
sgal11=abs(sgal11);
sgal11=sgal11./max(sgal11);
plot(w,sgal11);
grid;
title('GA ile Genlik Cevabı');
xlabel('w');
ylabel('Genlik');
    end  end
```

Şekil 5. 7. Alçak geçiren filtrenin katsayılarını üreten kod parçacıkları

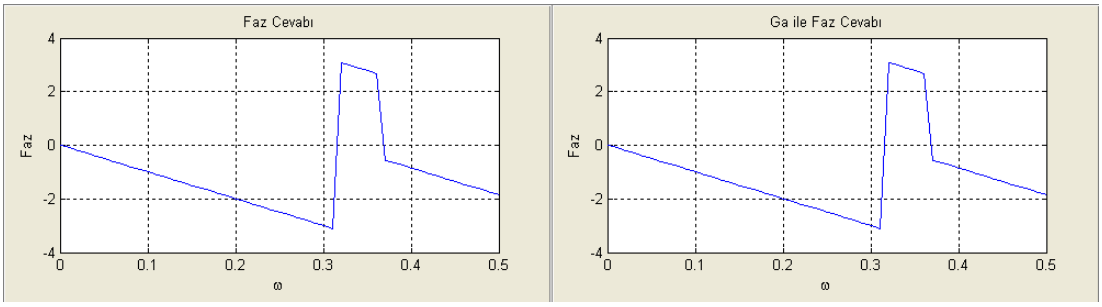
Katsayılar hesaplanmasında kullanılan kesim frekansı örnekleme frekansı ve alt kesim frekansı kullanılarak elde edilir. Bu değer program içerisinde hesaplandığı için

ekrana yansıtılmamıştır. Simetrik katsayılar kullanılarak genlik cevaplarını hesaplanarak grafikleri elde edilir. Genlik cevapları elde edilirken daha önceden belirlenen ω frekans değerlerine göre çizdirilir.

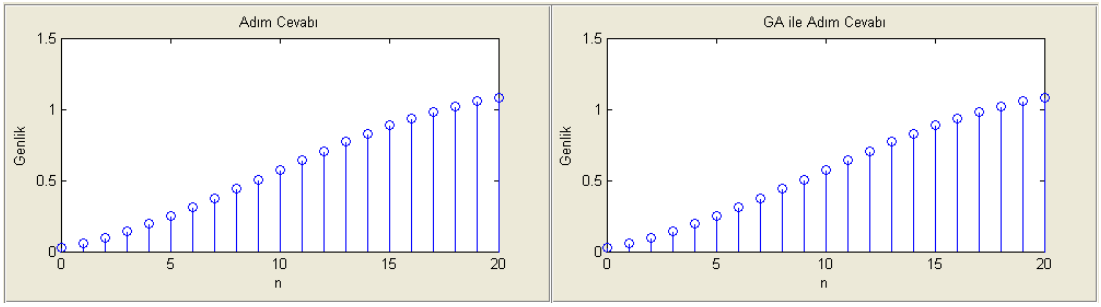
Tasarlanan örnekte kesim frekansı(ω_c) 0.2125'dır. Kesim frekansına kadar olan bütün frekans değerlerini geçirdiği grafikten görülmektedir. Şekillerde görülen grafikler genlik cevabı(db), faz cevabı, adım cevabı, dürtü cevabı ve filtrenin kutupları görülmektedir.



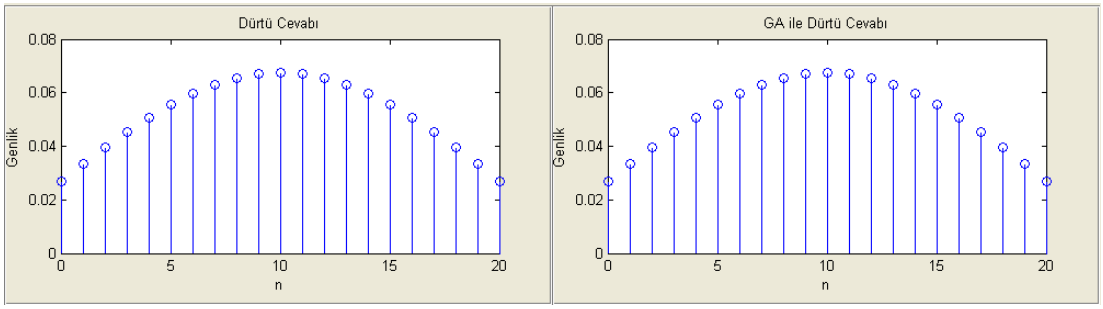
Şekil 5. 8. Alçak geçiren filtre genlik cevabı(db)



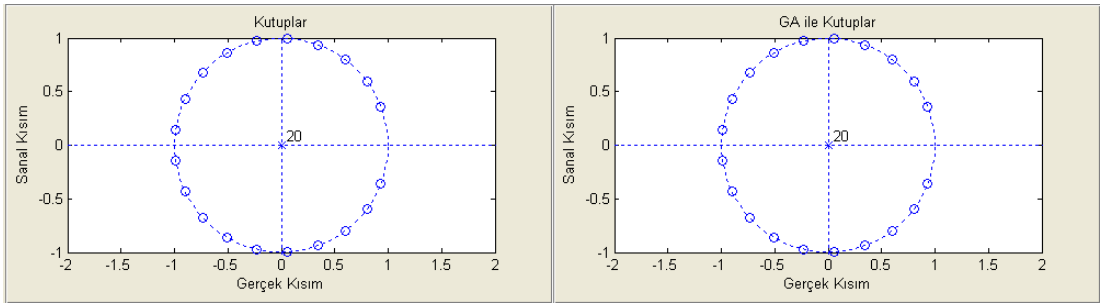
Şekil 5. 9. Alçak geçiren filtre faz cevabı



Şekil 5. 10. Alçak geçiren filtre adım cevabı

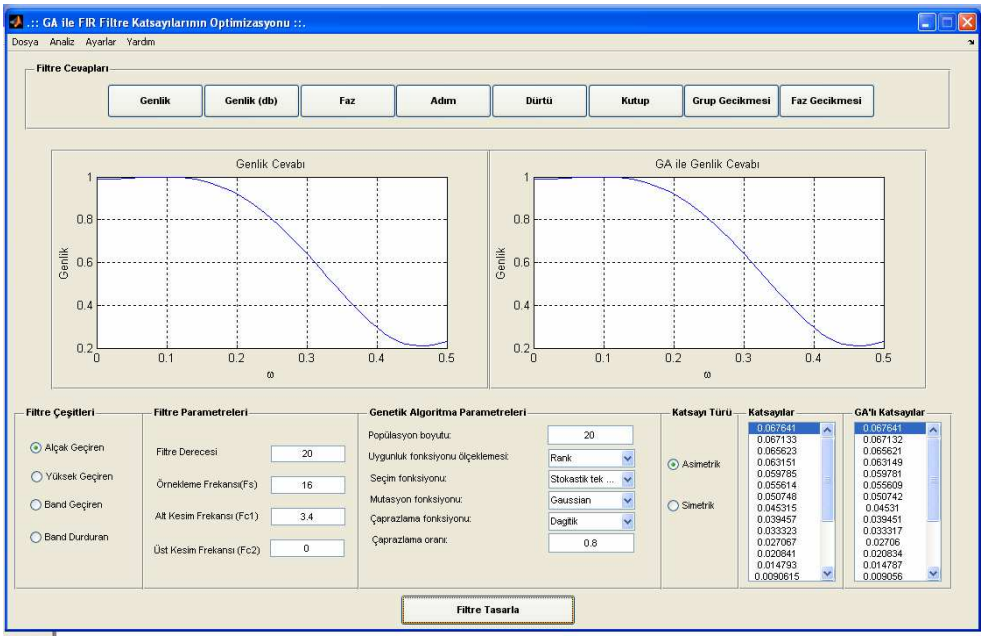


Şekil 5. 11. Alçak geçiren filtre dürtü cevabı



Şekil 5. 12. Alçak geçiren filtre kutupları

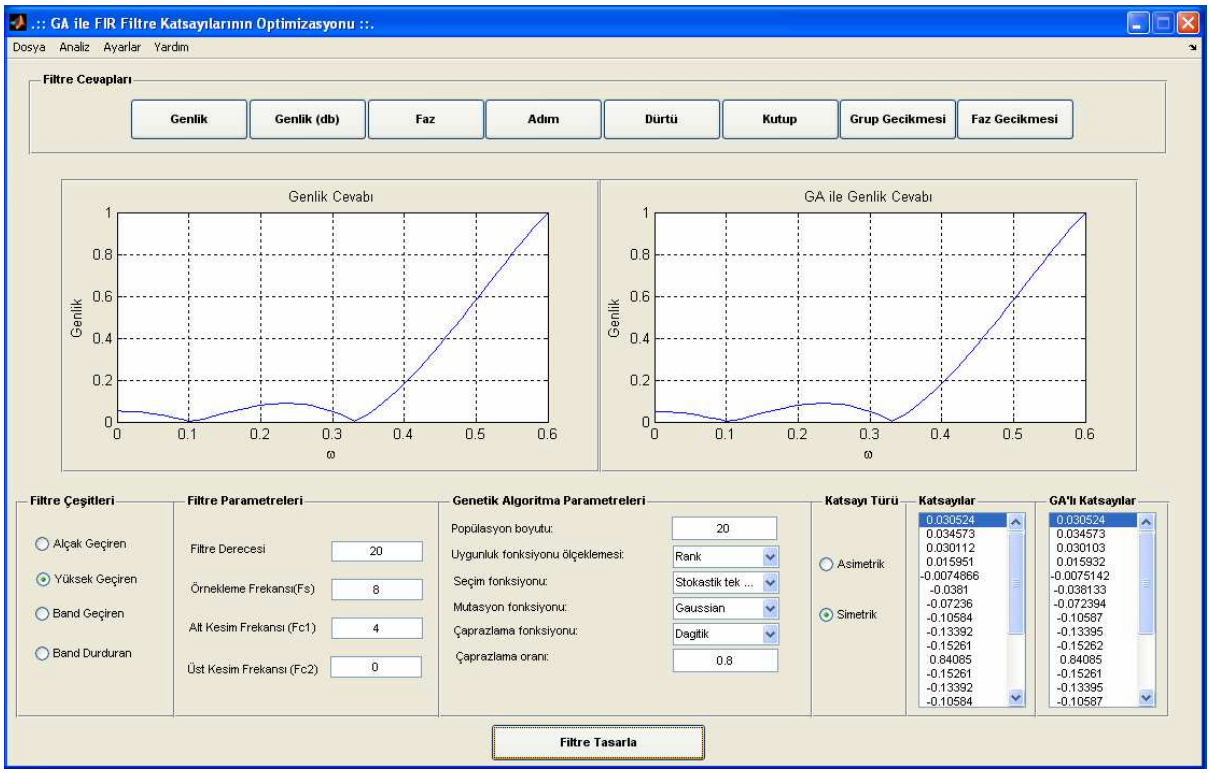
Eşitlik 2.13'deki denklem ile ifade edilen alçak geçiren FIR(sonlu dürtü cevabı) filtrenin asimetrik katsayılarını elde etmek içinde kullanılabilir. Diğerinden farklı olarak tasarım alanından katsayı türü asimetrik olarak seçilmesi gerekir.



Şekil 5. 13. Asimetrik alçak geçiren filtrenin katsayıları optimizasyonu

Eşitlik 2.15'deki denklem ile ifade edilen simetrik yüksek geçiren FIR(sonlu dürtü cevabı) filtrenin katsayılarını elde etmek için kullanılır. Bu denklemleri kullanarak tasarlanan arayüz aracılığıyla filtre tipi yüksek geçiren seçilir, katsayı türü olarak simetrik ve filtre parametreleri olarak filtrenin derecesi 20, örnekleme frekansı 8 kHz, üst kesim frekansı 4 kHz olarak ayarlanmıştır. Genetik Algoritma parametreleri varsayılan olarak kullanılmıştır. Parametre girişleri yapıldıktan sonra "Filtre Tasarla" butonuna tıklanarak simetrik olan ideal ve genetik algoritma kullanılarak elde edilmiş katsayılar ve genlik cevapları elde edilirken daha önceden belirlenen ω frekans değerlerine göre çizdirilir.

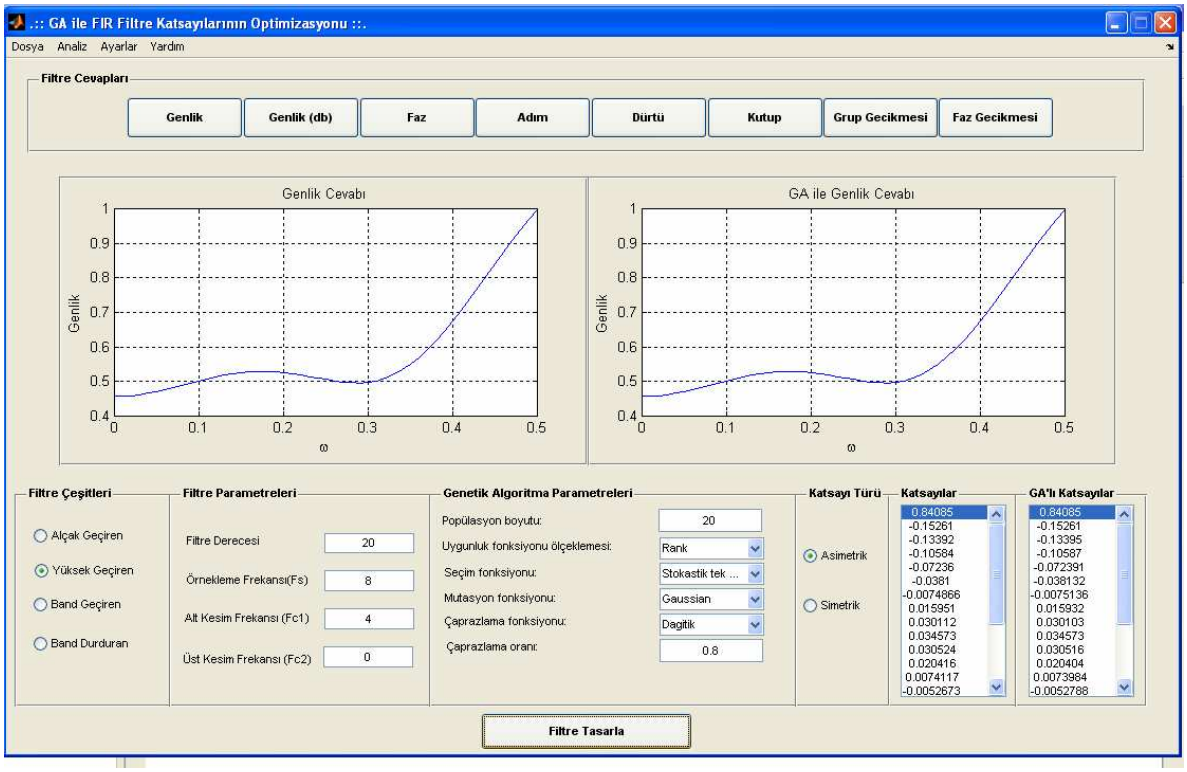
Katsayılar hesaplanmasında kullanılan kesim frekansı örnekleme frekansı ve üst kesim frekansı kullanılarak elde edilir. Bu değer program içerisinde hesaplandığı için ekrana yansıtılmamıştır. Tasarlanan örnekte kesim frekansı(ω_c) 0.5'dır. Kesim frekansına kadar olan frekans değerlerini geçirdiği grafikten görülmektedir.



Şekil 5. 14. Simetrik yüksek geçiren FIR filtrenin katsayıları optimizasyonu

Şekil 5.14'de görülen "Katsayılar" ve "GA'lı Katsayılar" kısmına bakıldığında katsayılar arasındaki farkın çok az olduğu görülmektedir. Sonuç genetik algoritma kullanılarak elde edilen katsayılar optimuma en yakın sonuçlardır. Bu katsayılar ile elde edilen genlik cevaplarında yüksek geçiren filtrenin temel yapısı olan belirli bir noktadan sonra olan frekanslar geçirdiği görülmektedir. Kesim frekansı (ω_c) 0.5 olduğu için bu noktadan sonra olan bütün frekanslar filtreden geçirilecektir.

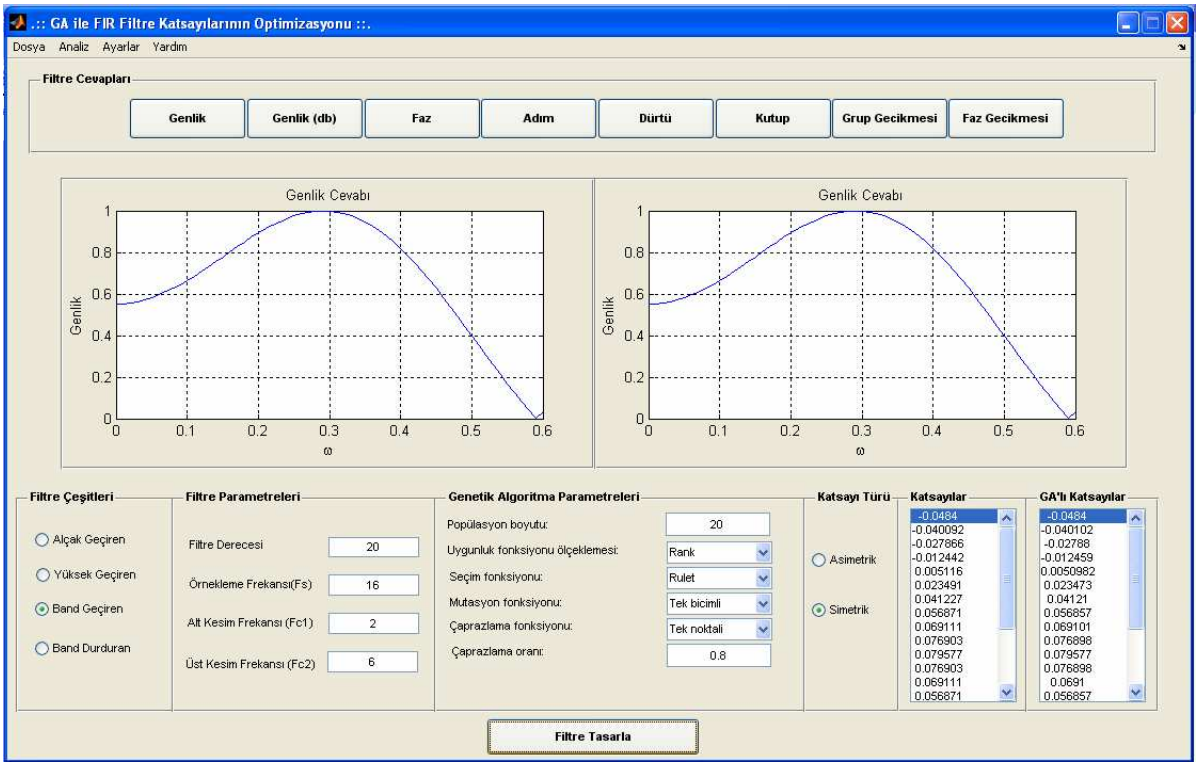
Şekil 5.15'te, Şekil 5.14'te uygulanan simetrik katsayılar kısmı değiştirilerek asimetrik yapılarak aynı parametrelerle çalıştırılır.



Şekil 5. 15. Asimetrik yüksek geçiren FIR filtrenin katsayıları optimizasyonu

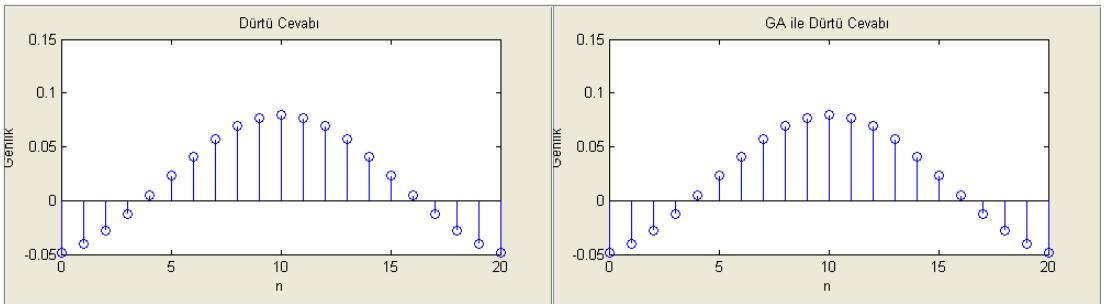
Şekil 5.15'de görüldüğü gibi genlik cevapları Şekil 5.14'deki gibi son frekans noktasında 1'e ulaştığı görülmektedir. Geçirme bandında aynı genişlikte olduğu görülmektedir. Geçiş bandı kısmında ise asimetrik özellikte olan filtrenin açısının daha geniş olduğu görülmektedir.

Eşitlik 2.17'deki denklem ile ifade edilen simetrik band geçiren FIR(sonlu dürtü cevabı) filtrenin katsayılarını ve cevaplarını elde etmek için kullanılır. Bu denklemleri kullanarak tasarlanan arayüz aracılığıyla filtre tipi band geçiren seçilir, katsayı türü olarak da simetrik katsayılar ve filtre parametreleri olarak da filtrenin derecesi 20, örnekleme frekansı 16 kHz, alt kesim frekansı 2 kHz, üst kesim frekansı 6 kHz olarak ayarlanmıştır. Genetik Algoritma parametreleri popülasyon boyutu "50", uygunluk fonksiyonun ölçeklenmesi "rank", seçim fonksiyonu "rulet", mutasyon fonksiyonu "tek biçimli", çaprazlama fonksiyonu "tek noktalı" olarak değiştirilmiştir. Parametre girişleri yapıldıktan sonra "Filtre Tasarla" butonuna tıklanarak simetrik olan ideal ve genetik algoritma kullanılarak elde edilmiş katsayılar ve genlik cevapları elde edilirken daha önceden belirlenen ω_1 ve ω_2 frekans değerlerine göre çizdirilir.(Şekil 5.16)



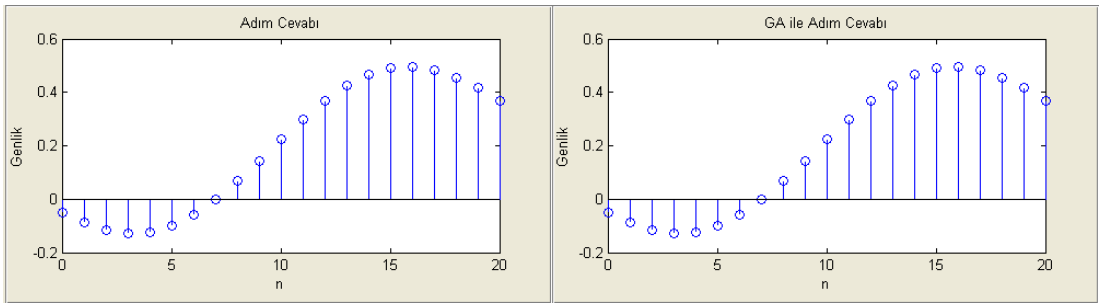
Şekil 5. 16. Simetrik band geçiren FIR filtrenin katsayıları optimizasyonu

Şekil 5.16'da genlik cevap çizimleri incelendiğinde iki nokta arasındaki frekansların geçirilmesine izin verilmiştir. Bu noktalar, alt kesim frekansı $\omega_{c1} = 2/16 = 0.125$ ve üst kesim frekansı $\omega_{c2} = 6/16 = 0.375$ 'dir.



Şekil 5. 17. Band geçiren FIR filtrenin dürtü cevabı

Şekil 5.17'da tasarlanan band geçiren filtrenin dürtü(impulse) cevabından katsayıların simetrik olduğu görülmektedir.

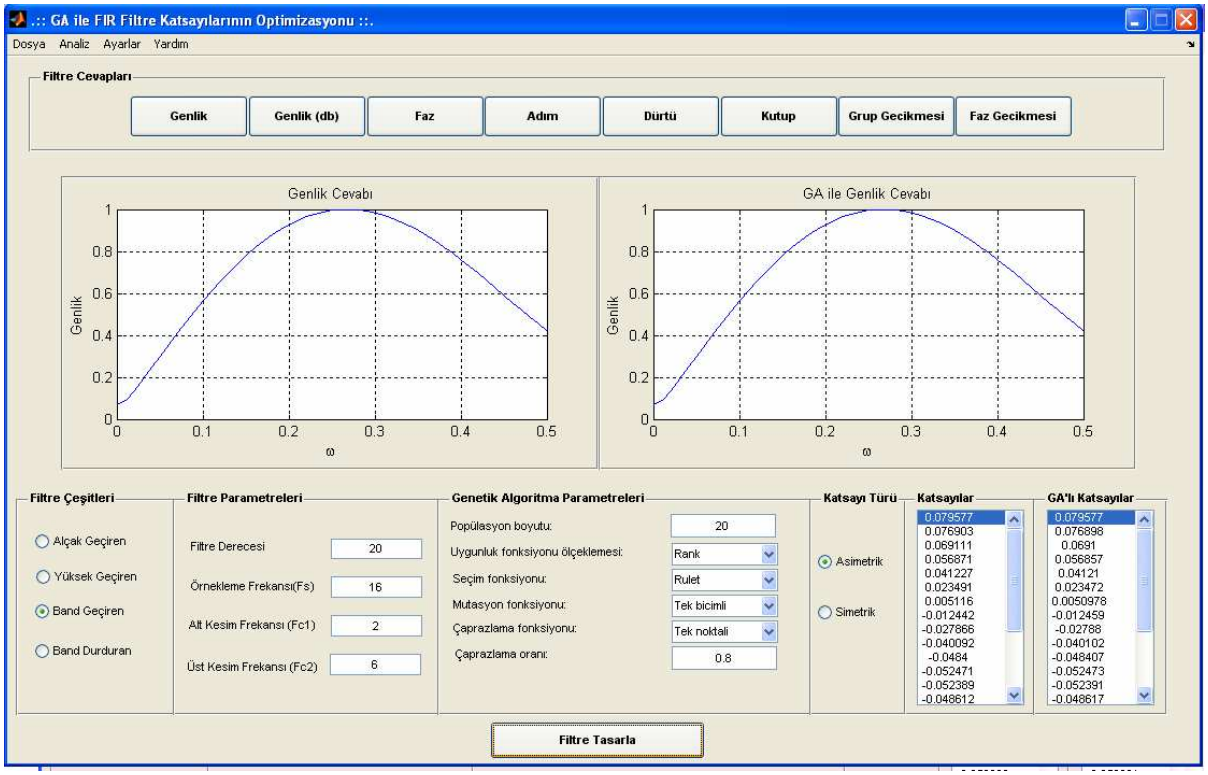


Şekil 5. 18. Band geçiren FIR filtrenin adım cevabı

Grafikler incelendiğinde hesaplanan değerler ve genetik algoritma ile elde edilen katsayılar arasında çok az bir fark vardır. Bu yüzden elde edilen grafiklerde de aynı sonuçlar elde edilir.

Şekil 5.19'da band geçiren FIR filtre tasarlanmıştır. Şekil 5.16'de tasarlanan band geçiren FIR filtrenin parametreleri kullanılmıştır. Fakat katsayı türü asimetrik olarak seçilmiştir.

Şekil 5.19'da genlik cevaplarından $\omega_{c1}(0.125)$ ve üst kesim frekansı $\omega_{c2}(0.375)$ arasındaki frekans değerleri geçirilmiştir.

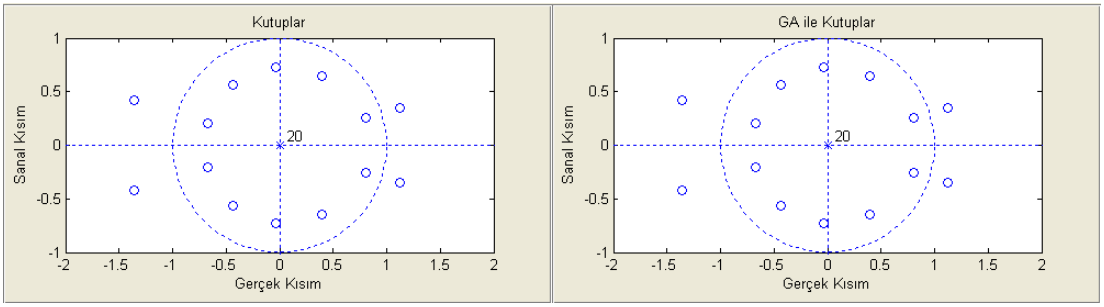


Şekil 5. 19. Asimetrik band geçiren FIR filtrenin katsayıları optimizasyonu

Eşitlik 2.19'daki denklem ile ifade edilen simetrik band durduran FIR (sonlu dürtü cevabı) filtrenin katsayılarını ve cevaplarını elde etmek için kullanılır. Bu denklemleri kullanarak tasarlanan arayüz aracılığıyla filtre tipi band durduran seçilir, katsayı türü olarak da simetrik katsayılar ve filtre parametreleri olarak da filtrenin derecesi 20, örnekleme frekansı 16 kHz, alt kesim frekansı 2 kHz, üst kesim frekansı 6 kHz olarak ayarlanmıştır. Genetik Algoritma parametreleri popülasyon boyutu "20", uygunluk fonksiyonun ölçeklenmesi "oransal", seçim fonksiyonu "turnuva", mutasyon fonksiyonu "adaptif", çaprazlama fonksiyonu "iki noktalı" olarak değiştirilmiştir. Parametre girişleri yapıldıktan sonra "Filtre Tasarla" butonuna tıklanarak simetrik olan ideal ve genetik algoritma kullanılarak elde edilmiş katsayılar ve genlik cevapları elde edilirken daha önceden belirlenen ω_1 ve ω_2 frekans değerlerine göre çizdirilir.(Şekil 5.20)

Şekil 5.16'da genlik cevap çizimleri incelendiğinde iki nokta arasındaki frekansların geçirilmesine izin verilmiştir. Bu noktalar, alt kesim frekansı $\omega_{c1} = 2/16 = 0.125$ ve üst kesim frekansı $\omega_{c2} = 6/16 = 0.375$ 'dir.

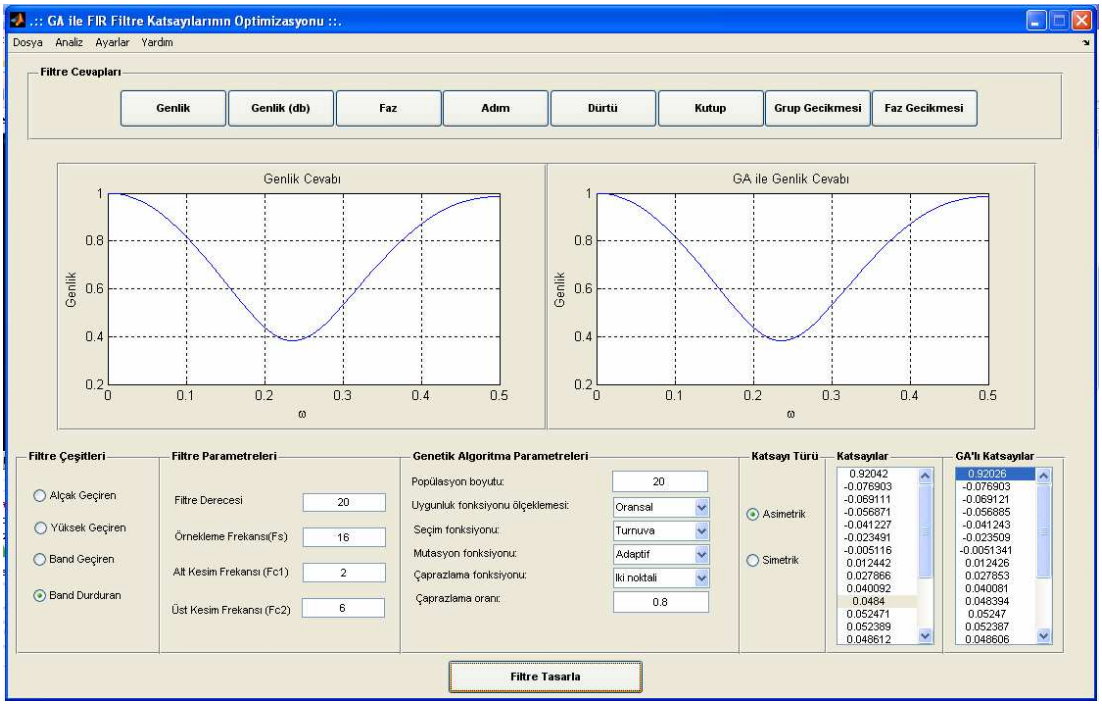
Şekil 5. 20. Band durduran FIR filtrenin simetrik katsayılarının optimizasyonu



Şekil 5. 21. Band durduran FIR filtrenin kutupları

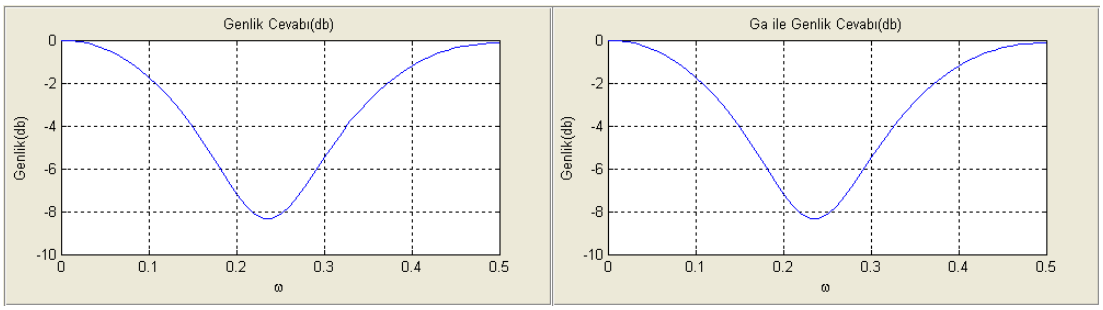
Şekil 5.21 incelendiğinde katsayılarda 21 olduğu için 21 tane kutup değeri olduğu görülmektedir.

Şekil 5.22'de asimetrik band durduran FIR filtre katsayıları tasarlanmıştır. Şekil 5.21'de tasarlanan band durduran FIR filtrenin parametreleri kullanılmıştır.

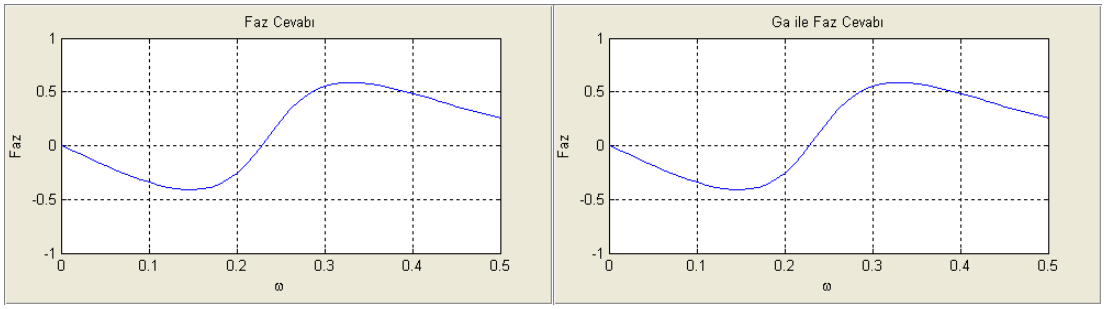


Şekil 5. 22. Asimetrik band durduran FIR filtrenin katsayıları optimizasyonu

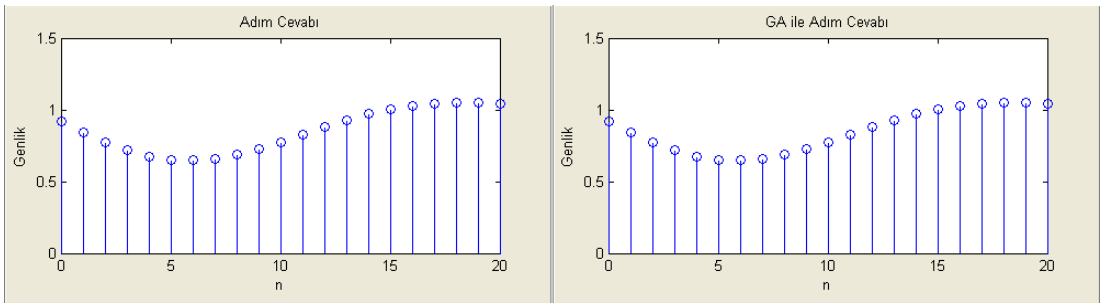
Şekil 5.22'de band durduran filtre tasarlanmıştır. Kesim frekansları arasındaki frekanslar geçirilmiştir. Şekil 5.20 ile karşılaştırıldığında şekil 5.22'deki genlik cevabının genlik cevaplarını daha düzgün bir şekilde çıkardığı görülmektedir. İstenen sonuç en yakın cevabı asimetrik katsayılar kullanılarak gerçekleştirilmiştir.



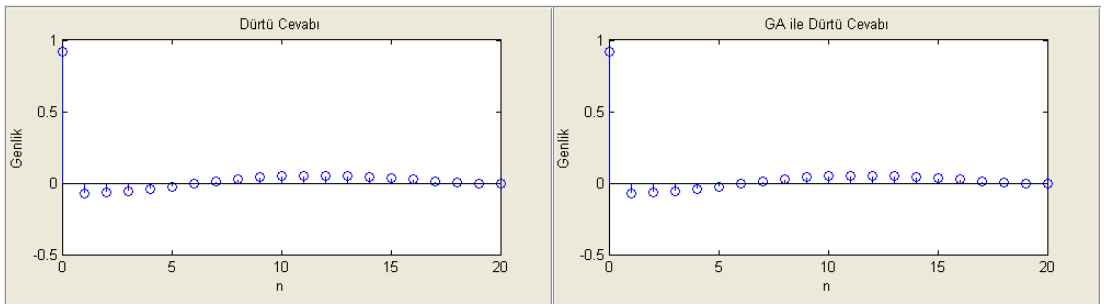
Şekil 5. 23. Band durduran FIR filtrenin genlik cevabı(db)



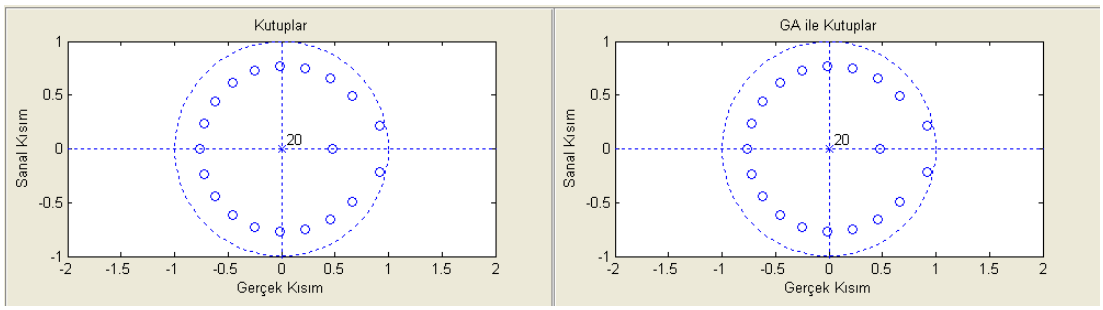
Şekil 5. 24. Band durduran FIR filtrenin faz cevabı



Şekil 5. 25. Band durduran FIR filtrenin adım cevabı



Şekil 5. 26. Band durduran FIR filtrenin dürtü cevabı



Şekil 5. 27. Band durduran FIR filtrenin kutupları

5.3. ASP.NET ve MATLAB Web Figure ile Hazırlanan Arayüzü

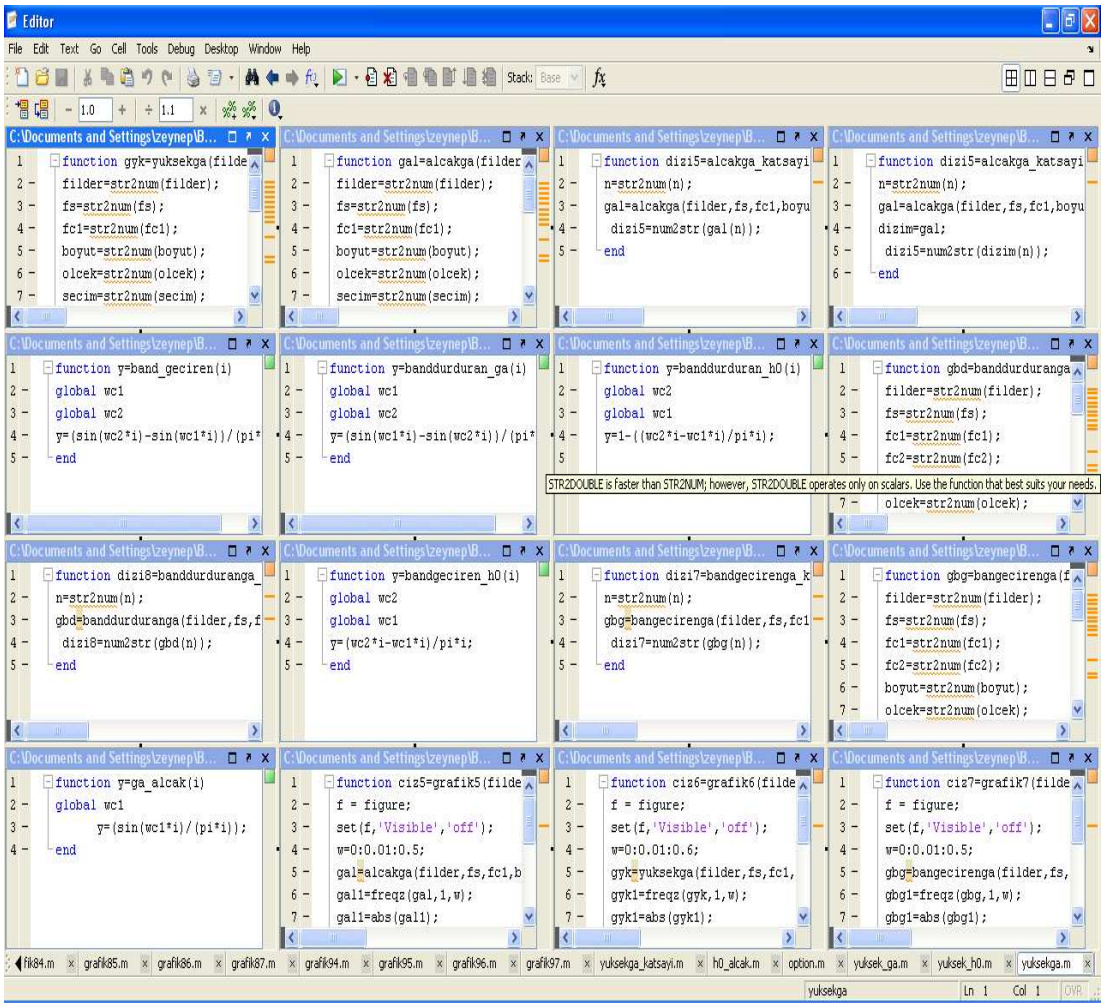
5.3.1. MATLAB BUİLDER NE ve MATLAB Web Figure

MATLAB programının içinde MATLAB Builder NE ve MATLAB Compiler araçları bulunur. MATLAB Builder NE, Matlab Compiler'ın bir alt ürünüdür ve Matlab fonksiyonlarına .Net programcılarının CLS-compliant dilleri olan C#, VB.Net ve C++ programlama dilleriyle ulaşmalarına olanak verir. Matlab Builder, Matlab kodlarını derleyerek Matlab fonksiyonlarını .Net metodlarına dönüştürür. Her Matlab Builder NE bileşeni bir veya daha fazla sınıf içerir. Her sınıf ise derleme işleminden önce eklenen Matlab fonksiyonları için arayüz oluşturur.

Matlab Builder NE veri dönüşümü, veri sıralama ve dizi biçimlendirme imkânı vererek, derlenmiş kod çağrıldığında Matlab fonksiyonları ve özelliklerinin esnek biçimde kullanılmasını sağlar. Bunun yanında Matlab Builder NE'de Matlab data tiplerinin desteklenmesi için MWArray.dll dosyasında tanımlı olan MWArray veri dönüşüm sınıfları bulunmaktadır. Kullanılan veri ve dizileri Matlab'a uygun biçime dönüştürmek ve Matlab'tan gelen verileri hazırlanan programda kullanabilmek için MWArray.dll dosyası uygulamaya referans olarak eklenmelidir.Matlab Builder NE'deki WebFigure özelliği ise Matlab figürlerini bir web sitesinde gösterebilme ve bu figürler üzerinde görsel olarak işlem yapabilme olanağı sağlamaktadır. Bu özellik son kullanıcılara, Matlab programı ve diğer araçlar olmadan grafiksel uygulamaları istedikleri bir yerden yalnızca web tarayıcı aracılığı ile internet üzerinden gerçekleştirme imkânı sağlamaktadır.

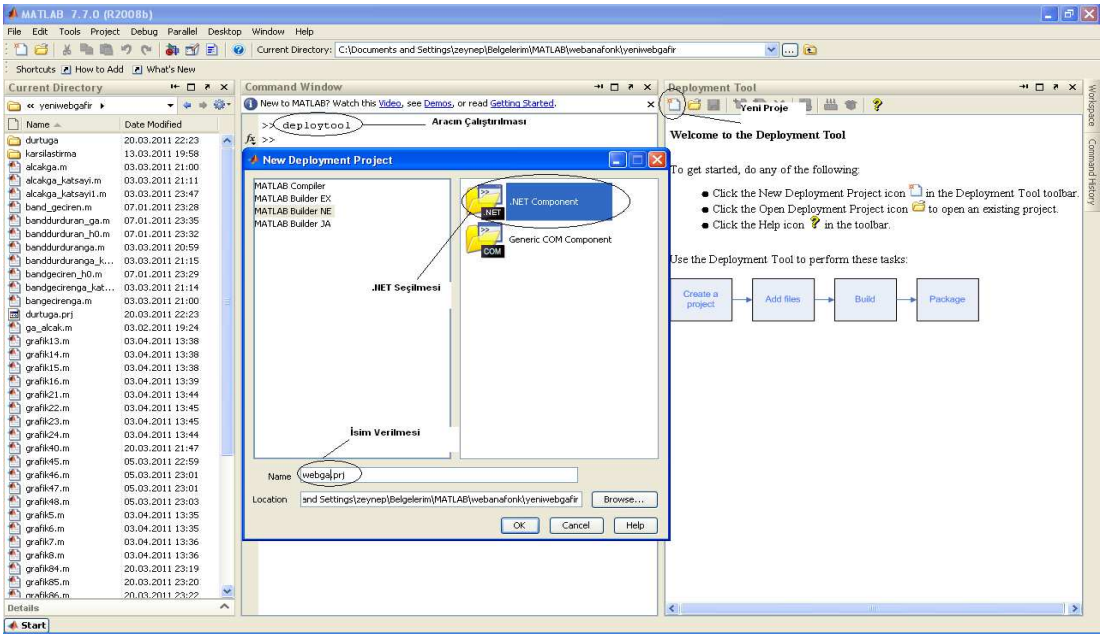
MATLAB Builder NE derleyicisini kullanarak .dll uzantılı .Net bileşenini oluşturmak için tasarımda kullanılan fonksiyonlarını işlemlere göre ayrı bir şekilde .m dosyası içerisinde tanımlanmalıdır. Arayüzde web figure aracı kullanılacağı için fonksiyonun içerisine figure komutu eklenir ve figure olarak grafiksel sonuç alınabilir. Şekil 5.27'de Web Figure için hazırlanan fonksiyonlar bulunmaktadır. Fonksiyonların içerisinde figure komutu kullanımı görülmektedir. FIR filtreler ideal

ve genetik algoritma kullanılarak elde edilen katsayılar ile gerçekleştirilmiştir. Şekil 5.28'de sadece genetik algoritma ile ilgili fonksiyonlar görülmektedir.



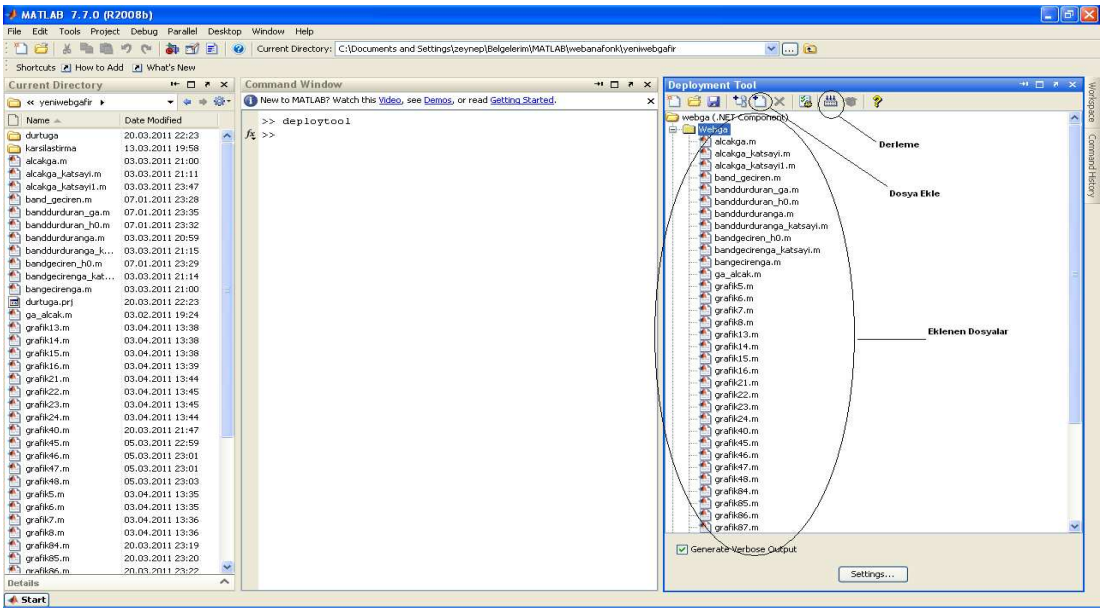
Şekil 5. 28. Derlenmek üzere oluşturulan .m dosyaları

.m dosyalarının derleme işlemini gerçekleştirebilmesi için MATLAB da bulunan geliştirme aracı olan "deploytool" aracı ile gerçekleştirilir. Bu araca ulaşmak için komut satırına deploytool komutu yazılır. Şekil 5.29'da görüldüğü gibi ilk önce derlenecek yeni bir component oluşturulur, yeni bir componet oluşturmak için seçili olan ikon seçilir.



Şekil 5.29. Geliştirme aracı (Deployment Tool)

Oluşturulan componente .m dosyalarını eklemek için Şekil 5.29' da gösterilen add file ikonuna tıklanarak dosyalar sınıfa eklenir ve derleme ikonuna tıklanarak derleme işlemi gerçekleştirilir.

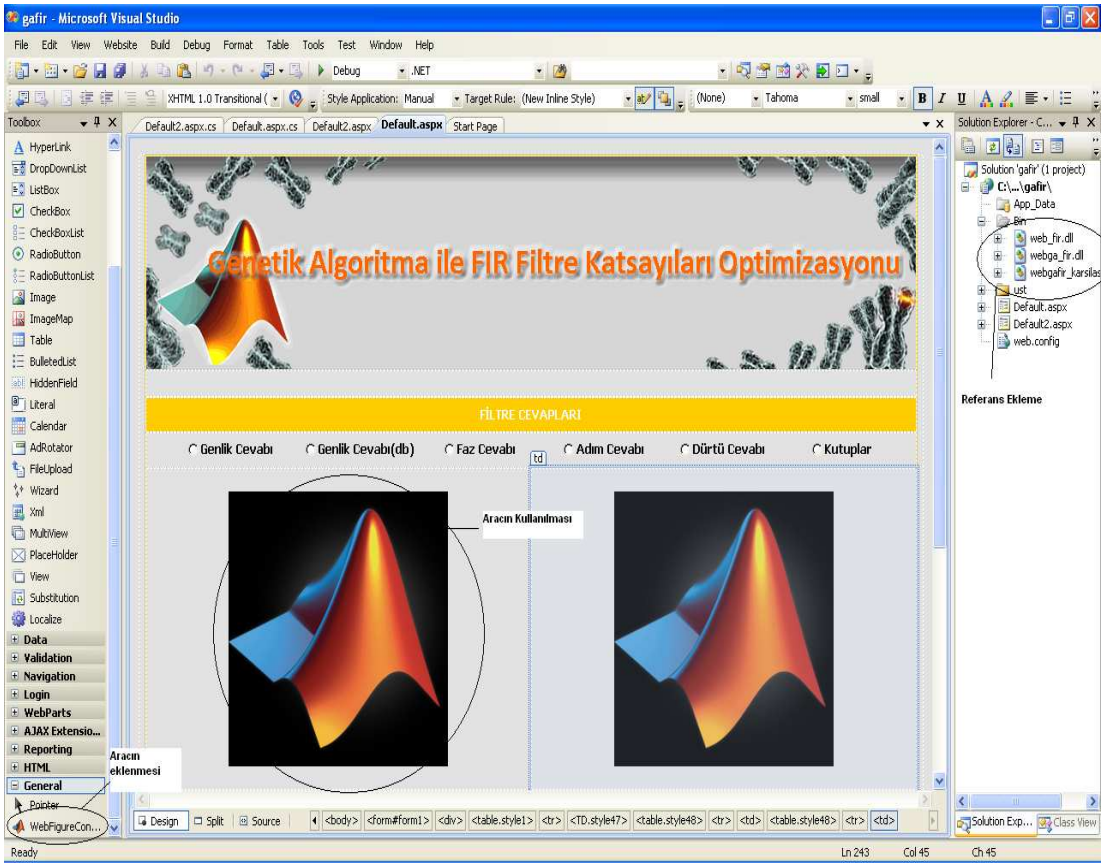


Şekil 5. 30. Geliştirme aracına (Deployment Tool) dosya eklenmesi

Matlab web figure aracını kullanmak için web arayüz alanına Şekil 5.31'deki gibi tasarımda kullanılan "Visual Web Developer" programında yapılacak ilk işlem "Web Figure Control" aracının araç kutusuna eklenmesidir.

İkinci işlem olarak çalışma alanına "web figure control" aracı sürüklenerek bırakılır. Web figure ait özellikler panelinden "name" kısmına uygulamada kullanılacak isim yazılır ve "scope" kısmından "application" seçeneği seçilir.

Üçüncü işlem olarak da matlab da hazırlanan .Net bileşeni uygulamada kullanılabilmesi için referans olarak eklenmelidir.



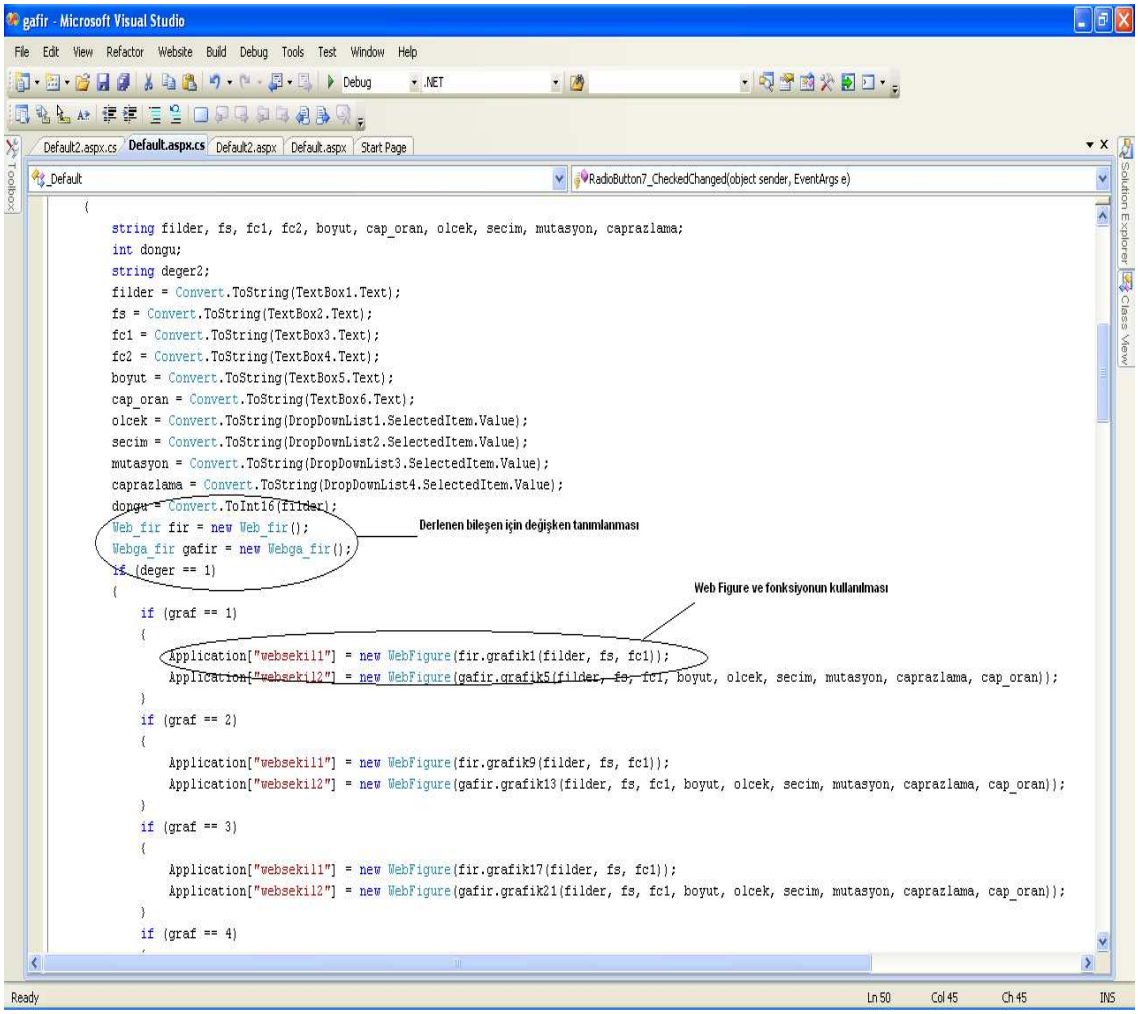
Şekil 5.31. Web arayüzü tasarım ekranı ve Web Figure kullanımı

Son aşama olarak Şekil 5.31'deki kod ekranına geçirilerek Matlab da gerçekleştirilmiş olan .dll içerisinden fonksiyonlar çağrılarak kodlar yazılır. .Net bileşenlerinin kullanılabilemesi için "using" komutu ile çağrılır. Bu çağırımlarda matlab fonksiyonlarının arayüzde kullanılabilmesi için aşağıdaki komutlar

eklenmelidir.

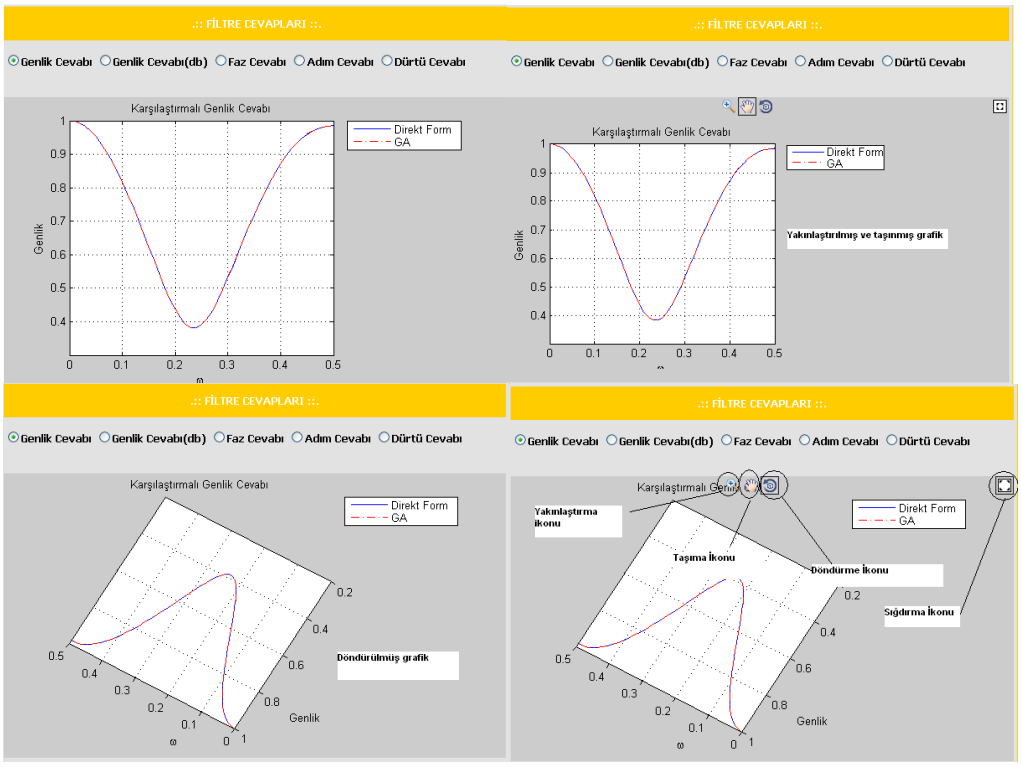
```
using MathWorks.MATLAB.NET.WebFigures;
using MathWorks.MATLAB.NET.Arrays;
using web_fir;
using webga_fir;
```

Derlenen bileşene ait yeni bir değişken tanımlanır. Şekil 5.32'de "fir" ve "gafir" değişkene ait olarak "grafik1" ve "grafik5" gibi oluşturulan fonksiyonlar .Net bileşeninde tanımlı olan web figure de gösterilecek fonksiyonlar bulunmaktadır. Web figure kontrol aracının kullanılabilmesi için Şekil 5.32'de gösterilen komutlarla kullanılması gereklidir.



Şekil 5. 32. Kod ekranı ve oluşturulan .Net bileşeninin kullanımı

Arayüz de grafiksel sonuçlar için kullanılan MATLAB Web Figure aracının özellikleri Şekil 5.33'de görülmektedir. Bu araç sayesinde elde edilen grafikler döndürme ikonu ile üç eksende döndürülmektedir, yakınlaşma ikonu ile grafiklerin istenen noktasına yakınlaştırabilmekte ve taşıma ikonu ile Web Figure çevresi istenen yere taşınabilmektedir. Çerçeveye sığdırma ikonu ile de grafiği eski haline döndürülebilir.



Şekil 5. 33. Web Figure özellikleri

5.3.2. Tasarlanan web arayüzü

Web arayüzü tasarımında Microsoft Visual Web Developer 2008 programını kullanılmıştır. Program aracılığıyla görsel tasarımlar geliştirmenin yanı sıra veri tabanı bağlantıları, hazır ajaxlar web uygulamalarında kolaylık sağlayacaktır. Bu tez çalışmasındaki web arayüz tasarımında ASP.NET tabanlı ve C# dili kullanılarak yapılmıştır. Hazırlanan program da iki farklı arayüz bulunmaktadır. İlk arayüzde MATLAB GUI'de gerçekleştirilen arayüze(Şekil 5.4) benzetilmiştir. Web arayüzü çalıştırıldığında ekrana Şekil 5.32'deki matlaba benzetilmiş arayüz gelmektedir.

Şekil 5. 34. Tasarlanan 1. Arayüz

Şekil 5. 35. Tasarlanan 2.Arayüz

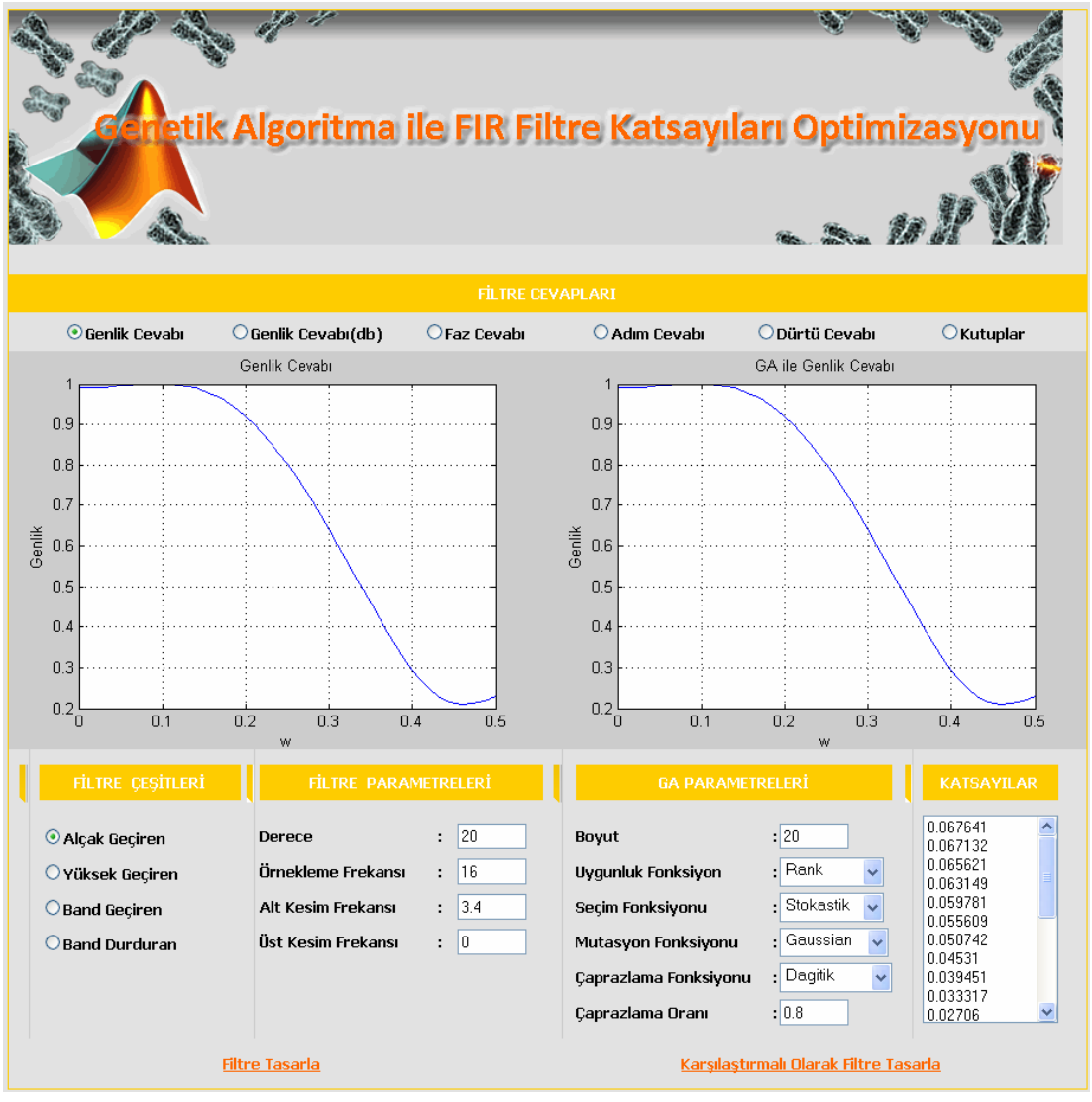
Tasarlanan bu arayüz de filtrenin çeşidinin seçilmesi için seçenekler, filtre parametrelerinin ve genetik algoritma parametrelerinin girilmesini sağlayan metin kutuları ve filtre cevaplarının seçilebileceği radio butonları bulunmaktadır.

Bu iki tasarım arasındaki ilk fark Şekil 5.34'deki tasarım da sadece genetik algoritma kullanılarak katsayılar elde edilir, Şekil 5.35'deki tasarımda hem hesaplanan katsayılar hem de genetik algoritma kullanılarak elde edilen katsayıların sonuçlarını elde edilebilir. Tasarımlar arasındaki diğer bir fark ise grafiksel sonuçlar da bulunmaktadır. Şekil 5.33'deki tasarımda grafiksel ortamı sağlayacak iki web figure var ilen Şekil 5.34'deki tasarımda ise bir web figure bulunmaktadır.

Grafiksel sonuçlar ve katsayıların hesaplanması için son aşama olan "Filtre Tasarla" ve "Karşılaştırmalı Olarak Filtre Tasarla" butonuna basıldığında analiz işlemi başlar ve sonuçlar elde edilir.

ASP.NET ve MATLAB Web Figure ile hazırlanan arayüzü ile MATLAB GUI ile hazırlanan arayüz arasındaki tek fark asp.net ile hazırlanan sayfada katsayılar standarttır yani simetrik değildir sadece asimetrik katsayıların hesaplaması gerçekleştirilebilir.

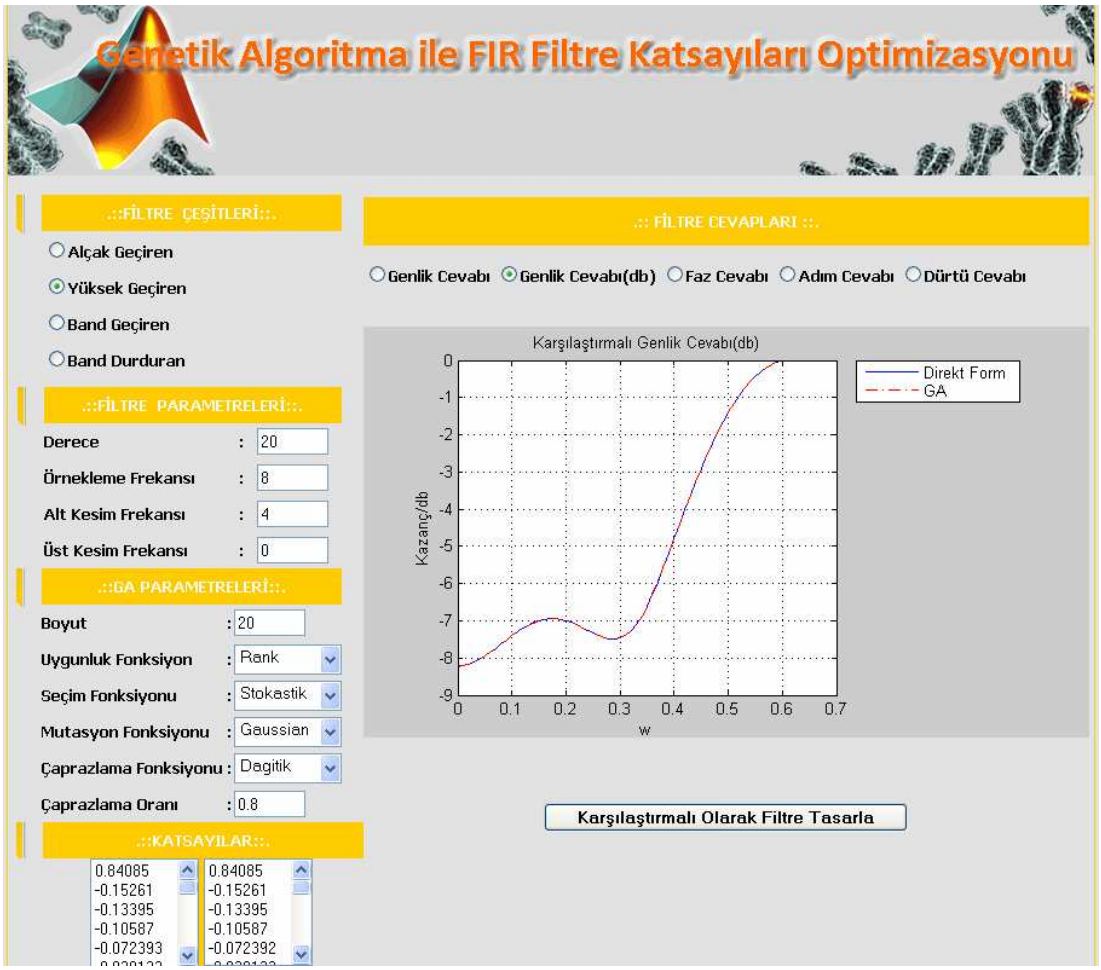
Eşitlik 2.13'deki denklem ile ifade edilen alçak geçiren FIR filtrenin katsayılarını elde etmek için kullanılır. Bu denklemleri kullanarak tasarlanan 1. arayüz aracılığıyla filtre tipi alçak geçiren seçilir, filtrenin derecesi 20, örnekleme frekansı 16 kHz, alt kesim frekansı 3.4 kHz olarak ayarlanmıştır. Genetik Algoritma parametreleri varsayılan olarak kullanılmıştır. Parametre girişleri yapıldıktan sonra "Filtre Tasarla" butonuna basılarak Şekil 5.36'daki şekil elde edilir.



Şekil 5. 36. Alçak geçiren FIR filtrenin girilen parametreleri ve analiz sonuçları

Şekil 5.37'de karşılaştırmalı olarak tasarlamayı gerçekleştiren arayüz kullanılarak yüksek geçiren FIR filtresi tasarlanmıştır.

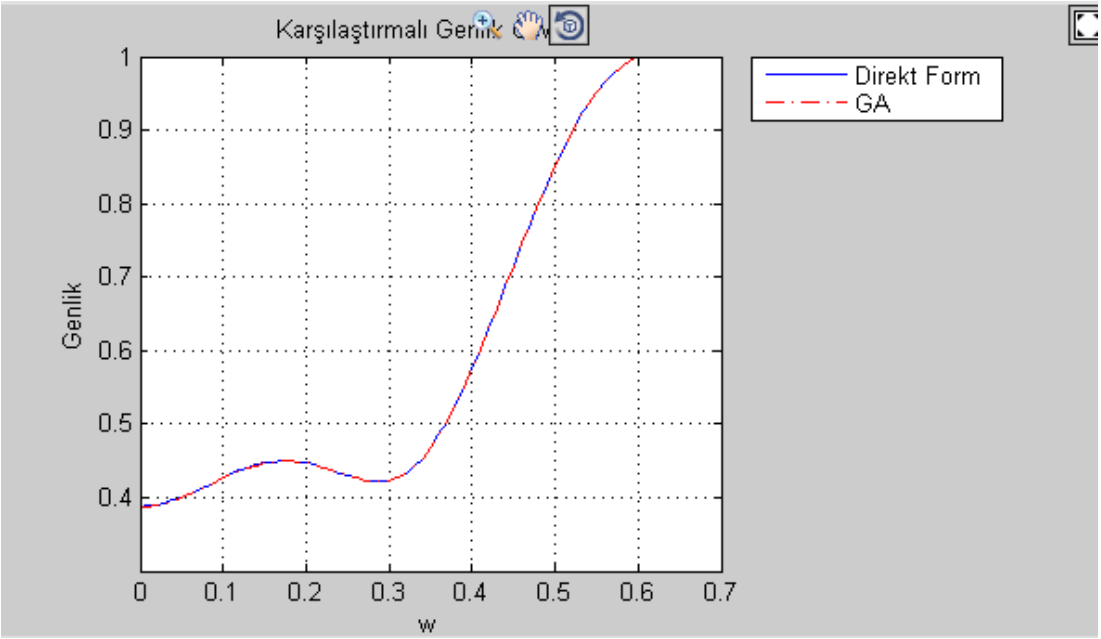
Bu denklemleri kullanarak tasarlanan arayüz aracılığıyla filtre tipi yüksek geçiren seçilir, filtre parametreleri filtrenin derecesi 20, örnekleme frekansı 8 kHz, üst kesim frekansı 4 kHz olarak ayarlanmıştır. Genetik Algoritma parametreleri varsayılan olarak kullanılmıştır. Parametre girişleri yapıldıktan sonra "Filtre Tasarla" butonuna tıklanarak ideal ve genetik algoritma kullanılarak elde edilmiş katsayılar ve seçilmiş olan genlik cevapları(db) karşılaştırmalı olarak elde edilir.



Şekil 5. 37. Karşılaştırmalı Genlik Cevabı(db) veren yüksek geçiren FIR filtresi

Şekil 5.37'de görüldüğü gibi genetik algoritma ile elde edilen katsayılar ile hesaplanan katsayıların değerleri birbirine çok yakın olduğu için elde edilen genlik cevapları da birbirine yakın grafikler elde edilir.

Şekil 5.38'de görülen grafik Şekil 5.36'de çalıştırılan programdan elde edilen genlik cevabıdır.

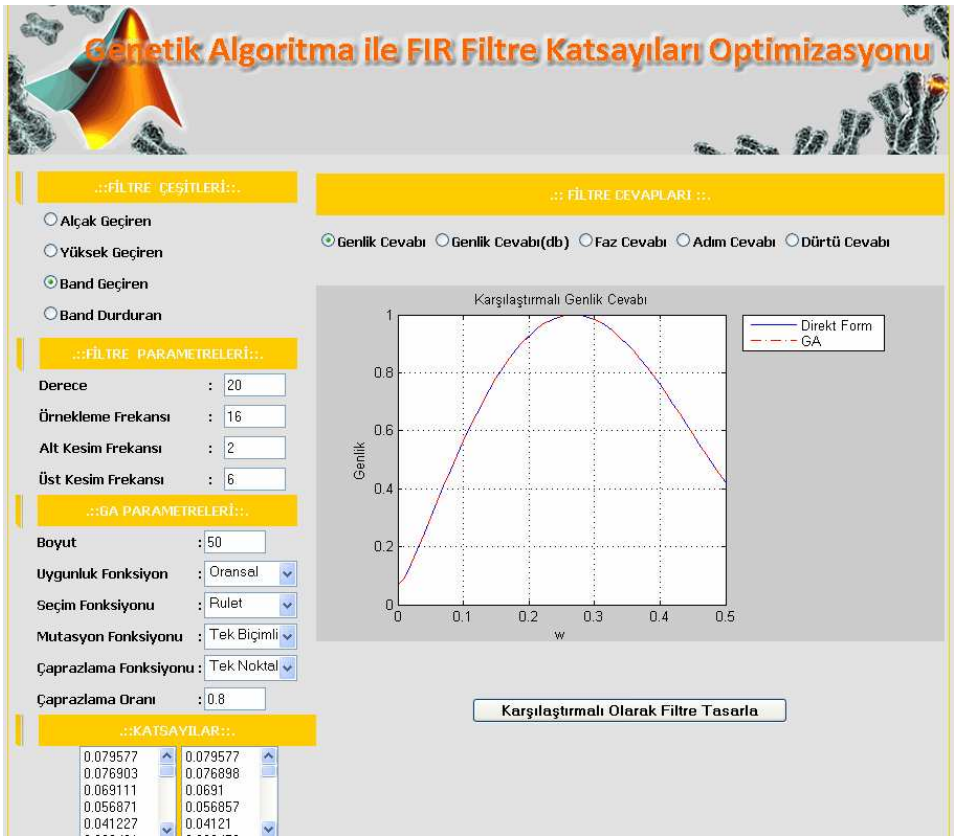


Şekil 5. 38. Yüksek geçiren FIR filtrenin genlik cevabı

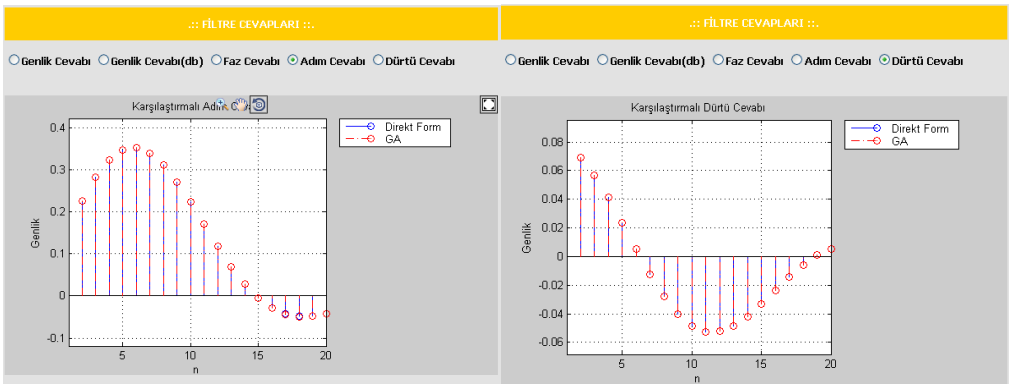
Şekil 5.39'da karşılaştırmalı olan tasarımı kullanılarak band geçiren FIR filtresi tasarlanmıştır.

Eşitlik 2.17'deki denklem ile ifade edilen band geçiren FIR(sonlu dürtü cevabı) filtrenin katsayılarını ve filtre cevaplarını elde etmek için kullanılır. Bu denklemleri kullanarak tasarlanan karşılaştırmalı arayüz aracılığıyla filtre tipi band geçiren seçilir. Filtre parametreleri olarak da filtrenin derecesi 20, örnekleme frekansı 16 kHz, alt kesim frekansı 2 kHz, üst kesim frekansı 6 kHz olarak ayarlanmıştır. Genetik Algoritma parametreleri popülasyon boyutu "50", uygunluk fonksiyonun ölçeklenmesi "rank", seçim fonksiyonu "rulet", mutasyon fonksiyonu "tek biçimli", çaprazlama fonksiyonu "tek noktalı" olarak değiştirilmiştir. Parametre girişleri yapıldıktan sonra "Karşılaştırmalı Olarak Filtre Tasarla" butonuna tıklanarak

simetrik olan ideal ve genetik algoritma kullanılarak elde edilmiş katsayılar ve genlik cevapları elde edilirken daha önceden belirlenen ω_1 ve ω_2 frekans değerlerine göre çizdirilir.(Şekil 5.15)



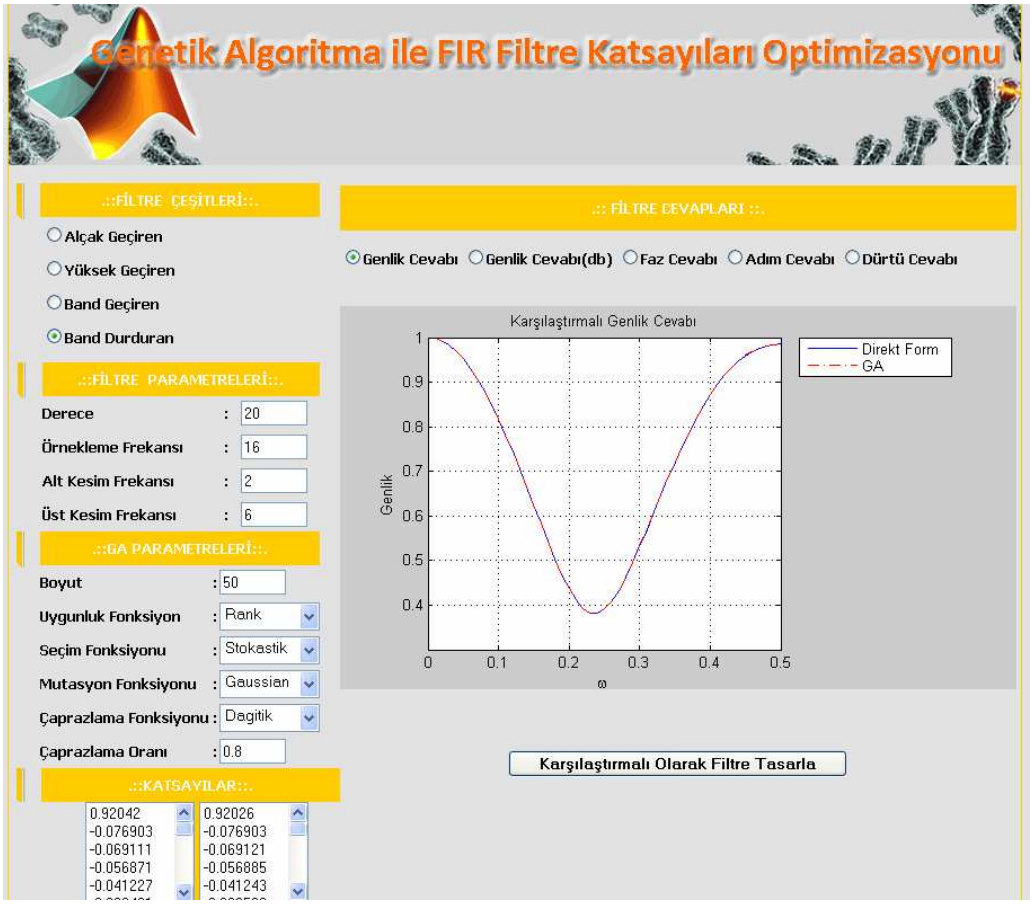
Şekil 5. 39. Karşılaştırmalı olarak genlik cevabı veren band geçiren FIR filtresi



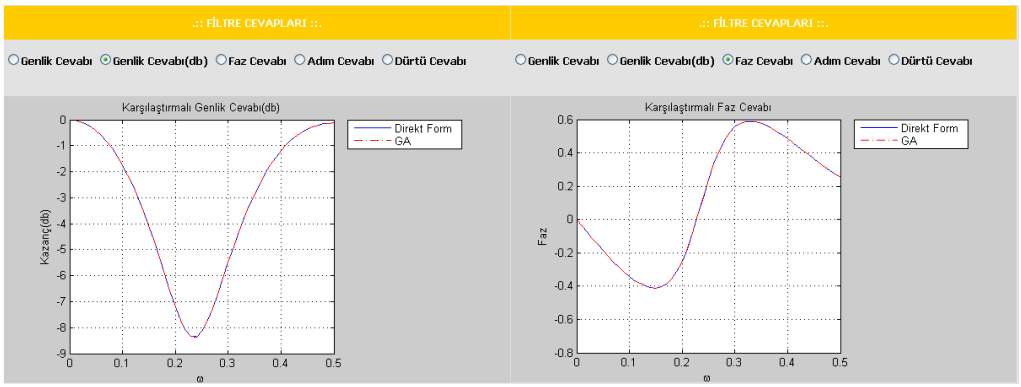
Şekil 5. 40. Karşılaştırmalı dürtü cevabı ve adım cevabını veren band geçiren FIR filtresi

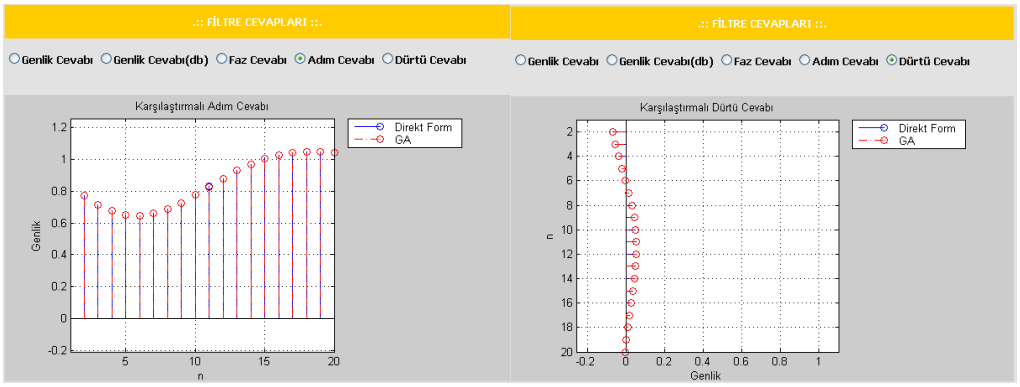
Şekil 5.41'de karşılaştırmalı olarak tasarımı gerçekleştirilen band durduran FIR filtresi tasarlanmıştır. Filtre ve genetik algoritma parametreleri şekil 5.41'de görülmektedir.

Bu tasarımda karşılaştırmalı genlik cevabı ve katsayılar elde edildiği görülmektedir.



Şekil 5. 41. Karşılaştırmalı olarak genlik cevabı veren band durduran FIR filtresi





Şekil 5. 42. Karşılaştırmalı olarak genlik cevabı(db), faz cevabı, adım cevabı ve dürtü cevabını veren band durduran FIR filtresi

Yukarıdaki şekiller analiz edildiğinde aynı tip filtreler ve parametreler için web arayüzü ve MATLAB GUI arayüzü ile elde edilen sonuçlar birbirinin aynısı olarak elde edilmiştir.

BÖLÜM 6. SONUÇ VE DEĞERLENDİRMELER

Sayısal sinyal işleme, sinyallerin bir sayısal veya sembol dizileriyle ve bu dizilerin işlenişleriyle ilgilidir. Sinyal işlemenin amacı belli bir sinyalin karakteristik parametrelerini belirleyebilmek ve kullanışlı bir sinyal elde edebilmektir.

Sayısal filtrelerle tümleşik halde bulunan sinyallerin ayrılmasını ve bozulmuş sinyallerin onarılması sağlanır. Sayısal filtreler sayısallaştırılmış analog sinyalleri kullanarak çalışmaktadır. Sayısallaştırma, analog sinyalin örneklenmesini gerektirir ve sayısal değere dönüştürülür.

Sayısal filtreler genellikle digital sinyal işlemcisi tarafından entegre halinde tasarlandıkları için bir değişim olması gerektiği zaman sadece yazılım kısmında değişiklik olması gereklidir oysa analog filtreler de devre tasarımındaki elemanların değişmesi gerekir bu da maliyet ve zaman bakımından filtreler arası farkı gösterir. Analog filtrelerin çalışma noktası ve kararlığı sıcaklığa bağlı olduğu halde sayısal filtreleri sıcaklık etkilemez. Sayısal filtreleri gerçekleştirirken analog filtreden daha fazla gerçekleştirilme algoritmasına sahiptir. Sayısal filtreler analog filtrelere göre daha düşük frekanslı sinyallerle çalışabilirler. Analog filtrelerin sayısal filtre göre üstünlükleri ise enerjiyi geçirme, frekans aralık sınırı verilmemesi gibi sayılabilir.

Sayısal filtreler(süzgeçler) dürtü yanıtlarına göre sonsuz dürtü yanıtlı filtre (Infinite Impulse Response (IIR)) ve sonlu dürtü yanıtlı (Finite Impulse Response (FIR)) filtre şeklinde ikiye ayrılmaktadırlar.

Tasarım aşamasında çıkışın ideale yakın olması istendiği için filtre derecesi artmakta ve filtrenin tasarımındaki çarpma ve toplama eleman sayısında artmalara neden olmaktadır. İstenen özelliklerde filtre tasarlayabilmek için pek çok yöntem bulunmaktadır. Filtre tasarımında (gerek IIR gerekse de FIR filtrede) temel amaç,

fonksiyonunu oluşturan pay ve payda katsayılarının hesaplanması işlemine dayanmaktadır.

Sezgisel hesaplama yöntemlerinden biri olan ve sonuca farklı arama noktalarından değer üreterek ulaşan GA, klasik hesaplama yöntemlerinin yetersiz kaldığı veya farklı çözüm yollarının üretilmesinin istendiği uygulamalarda fazlaca kullanılmaktadır.

Bu tez çalışmasında sonlu dürtü yanıtı (FIR) filtre çeşitlerini genetik algoritma yöntemi kullanılarak filtre katsayıları elde edilmiştir.

Bu iki yöntemden elde edilen katsayılar kullanılarak, filtrelerin genlik cevabı, faz cevabı, dürtü cevabı ve adım cevapları için karşılaştırılmıştır. Genetik algoritma ile katsayılar elde edilirken her çalıştırıldığında farklı sonuçlar elde edilmesine rağmen sonuçların birbirine çok yakın olduğu görülmektedir. Parametre değişikliğinden dolayı katsayıların hassasiyet oranlarının değiştiği incelenmiştir. Ayrıca programda tek bir filtre türü incelenmemiştir. Filtre çeşitleri olarak alçak, yüksek, band geçiren ve band durduran filtre türleri de incelenmiştir.

Bu yöntemleri incelemek için biri MATLAB GUI diğeri ise ASP.NET olmak üzere iki farklı arayüz tasarlanmıştır. Tasarımlarla birlikte, hesaplama güçlüğünü ortadan kaldıran ve işlemleri otomatikleştiren metodun kullanımında kolaylık sağlamaktadır.

MATLAB GUI ile gerçekleştirilen arayüzden elde edilen sonuçlar saklanabilir ve başka bir filtre çeşidine entegre edilebilir. ASP.NET ile hazırlanan web tabanlı arayüz ile çok sayıda kullanıcının kullanım olanağı verilebilir. Bu özellik kullanılarak teknik eğitimde teorik bilgilerin laboratuar ortamına benzetilerek geliştirilmesi ve simülasyonlarının gerçekleştirilmesinde fiziki koşulların eksikliği giderilebilir ve bilgisayar destekli eğitimin etkisi artırılabilir. Bu özellikler tasarımların avantajları arasında sayılabilir.

Arayüzlerin önemli bir dezavantajı olarak işlem yükünden kaynaklanan yavaş çalışma gösterilebilir. ASP.NET ile gerçekleştirilen web tabanlı tasarımda katsayılar

elde edilirken MATLAB ilişkili olduğundan dolayı beklenen süre artmaktadır. MATLAB GUI arayüzünün web tabanlı tasarımlara göre daha esnek olduğu görülmektedir.

Yapılan çalışmalara ek özellikler eklenerek iyileştirilebilir. Buna örnek olarak tasarımlara pencereleme tekniği de eklenebilir. Bu pencereleme tekniklerine örnek olarak dikdörtgensel, üçgensel, von hann, hamming, blackman gibi pencerelemeler örnek verilebilir ve sayısal filtre çeşitlerinden IIR filtre türü de eklenebilir.

KAYNAKLAR

[1] KEDİR-TALHA, M., GUETTOUCHE, M., and BOUSBİA-SALAH, A., Combination of a FIR filter with a genetic algorithm for the extraction of a fetal ECG, Proceedings of the International Conference on Circuits, Systems, Signals , 76-79, 2010

[2] BAYILMIŞ, C., Development of a web-based educational interface using MATLAB builder NE and web figure for digital modulation techniques, Computer Applications in Engineering Education. DOI 10.1002/cae.20427

[3] DEY, A., SAHA, A., SAHA, S., GHOSH, S., A Method of Genetic Algorithm (GA) for FIR Filter Construction: Design and Development with Newer Approaches in Neural Network Platform, International Journal of Advanced Computer Science and Applications, Vol. 1, No. 6, December 2010

[4] KAÇAR, S., ÇANKAYA, İ., Volterra Serileri Metodu ile Doğrusal Olmayan Sistemlerin Frekans Boyutunda Analizi İçin .Net Tabanlı Arayüz Tasarımı, Deü Mühendislik Fakültesi Fen Bilimleri Dergisi, Cilt: 12 Sayı: 3 sh., 87-102, Ekim 2010

[5] GOYAL, S., RAİNA, J., Design of Low Power FIR Filter Coefficients Using Genetic Algorithm, Vol. 1, No. 2, pp. 1-5, July-December 2010

[6] KAÇAR, S., ÇANKAYA, İ., BAYILMIŞ, C., ÇAKIROĞLU, M., Kablosuz Algilayıcı Ağlar İçin Matlab Builder Ne ve Matlab Webfigure ile Asp.Net Tabanlı Web Arayüzü Tasarımı, e-Journal of New World Sciences Academy, Volume: 4, Number: 4, 360-370, 2009

[7] KAYA, T., İNCE, M., Genetik Algoritma Tabanlı Bilineer Dönüşümü Kullanarak Sayısal Bir Süzgecin Tasarımı ,Uluslararası İleri Teknolojiler Sempozyumu (IATS'09), Karabük, Türkiye, 13-15 Mayıs 2009

[8] ÇANKAYA, İ., VATANSEVER, F., AKGÜN, D., Rlc Filtre Devrelerinin Eğitimine Yönelik Grafiksel Arayüz Tasarımı, 5. Uluslararası İleri Teknolojiler Sempozyumu (IATS'09), Karabük,Türkiye 13-15 Mayıs 2009

[9] AHMAD, A., Design of Digital Filters Using Genetic Algorithms, Doktora, University of Victoria, Electrical and Computer Engineering, 2010

[10] KAYA, T., İNCE, M.,Genetik Algoritmaların Aktif Filtrelerde Kullanımı, ELECO-2008 Elektrik-Elektronik-Bilgisayar Mühendisliği Sempozyumu, 512-515, , Bursa, 26-30 Kasım 2008

[11] AGNIHOTRI, S., Design and Development of a Power Efficient Code for FIR Filter Coefficients Using Genetic Algorithms, THAPAR UNIVERSITY, Department of Electrical and Instrumentation Engineering, July – 2008

[12] KAYA, T., İNCE, M.C. Genetik Algoritmaların Butterworth Filtre Parametrelerinin Hesaplanmasında Kullanımı, Çukurova Üniversitesi Mühendislik-Mimarlık Fakültesi 30. Yıl Sempozyumu, Adana, 16-17 Ekim 2008

[13] KAYA, T., İNCE, M. C., Sayısal Filtre Katsayılarının Genetik Algoritma Yardımıyla Hesaplanması, Fırat Üniversitesi Fen ve Mühendislik Bilimleri Dergisi, Cilt: 20, Sayı: 3, 459-466, 2008.

[14] BARROS, A., STELLE, Á., LOPES, H., A Fır Fılter Desıgn Tool Usıng Genetıc Algorıthms, Anais Do XXXIV Cobenge, Ed. Universidade de Paso Fundo, Eylül 2006

[15] KAYA, T., Genetik Algoritma ile Sayısal Filtre Tasarımı, Yüksek Lisans, Fırat Üniversitesi Fen Bilimleri Enstitüsü, Elektrik-Elektronik Mühendisliği Ana Bilim Dalı, Fırat, 14.08.2006, 74, 2006

[16] TZENG, T., Design of 1-D Complex FIR Digital Filters by Genetic Algorithm Approach ,Proceedings of 2005 CACS Automatic Control Conference, Nov 18-19, 2005

[17] KARABOGA, N., CETİNKAYA, B., Design of Minimum Phase Dijital IIR Filters by Genetic Algorithm, Proceedings of the 6th Nordic Signal Processing Symposium - NORSIG 2004, Espoo, Finland, June 9 - 11, 2004

[18] TZENG, S.-T., Genetic algorithm approach for designing 2-D FIR digital filters with 2-D symmetric properties, Signal Processing 84, 1883–1893, 2004

[19] LİANG, L., AHMADİ, M., SİD-AHMED, M., WALLUS, K., Signal, Sytems and Computers, Conference Record of the Thirty-Seventh Asilomar Conference, 2043-2047 , 9-12 Nov. 2003

[20] REDMİLL, D.,BULL, D., Automated Design Of Low Complexity Fir Filters, The IEEE International Symposium on Circuits and Systems (ISCAS), 429 - 432 vol.5, 1998

[21] ARSLAN, T., HORROCKS, D.H., The Design of Analogue and Digital Filters Using Genetic Algorithms, The IEEE International Symposium on Digital and Analogue Filters and Filtering Systems, 2/1 - 2/5, 1995

[22] KARABOĞAN, N., Sayısal Filtre Katsayılarının Genetik Algoritma Kullanılarak Yuvarlatılması, Doktora, Erciyes Üniversitesi Fen Bilimleri

Enstitüsü, Elektronik Ana Bilim Dalı, Erciyes, 27.01.1994, 126, 1994

[23] SUCKLEY, D., Genetic algorithm in the design of FIR filters, IEE PROCEEDINGS-G, Vol. 138, No. 2, 1991

[24] STEVE, W., Analog and Digital Filter Design, Newnes Yayınevi, 450, 2002

[25] THEDE, L., Practical Analog And Digital Filter Design , Artech House Yayınevi,270, 2004

[26] INGLE V., PROAKİS, J., Digital Signal Processing Using MATLAB V.4, PWS Yayınevi , 420, 1996

[27] PUNSKAYA, E.,Design of FIR Filters ders notu.
Web Site: www-sigproc.eng.cam.ac.uk/~op205 (Erişim bilgi: Ocak 2011)

[28] MADİSETTİ, V., The Digital Signal Processing Handbook, CRC Yayınevi, 1776, 1997-12-29

[29] CORMEN, T., LEİSERSON, C., RİVEST, R., STEİN, C., Introduction to Algorithms, MIT Yayınevi, 1332, 2009

[30] MELANİE, M., An Introduction to Genetic Algorithms, MIT Yayınevi, 162, 1998

[31] ŞEN, Z., Genetik Algoritmalar ve En İyileme Yöntemleri, Su Vakfı Yayınevi, 142, 2004

[32] Matlab Builder NE 3 User's Guide, The Mathworks Inc., Ekim 2008

[33] MATLAB EQ: Background on Equalization Website.
http://cnx.org/content/m15655/latest/ (Erişim bilgi: Ocak 2011)

[34] TEKİN, A., ATA F., Sanal Bir Laboratuar: Asenkron Motorların Hız Denetimi, Fırat Üniv. Mühendislik Bilimleri Dergisi,22 (1), 73-82, 2010

[35] Matlab Com Builder User's Guide, The Mathworks Inc., Ekim 2002

[36] IRMAK, E., E-Öğrenme Ortamlari İçin Matlab Web Sunucu Kullanimi, Gazi Üniv. Müh. Mim. Fak. Der., Cilt 23, No 2, 495-506, 2008

ÖZGEÇMİŞ

Zeynep BATIK, 02.11.1988'de Sakarya' da doğdu. İlk, orta ve lise eğitimini Adapazarı'nda tamamladı. 2005 yılında Adapazarı Anadolu Kız Meslek Lisesi, Bilgisayar Bölümünden mezun oldu. 2005 yılında başladığı Sakarya Üniversitesi Teknik Eğitim Fakültesi Bilgisayar Sistemleri Öğretmenliği bölümünden 2009 yılında mezun oldu. 2009 yılında Sakarya Üniversitesinde Bilgisayar ve Elektronik Eğitimi bölümünde yüksek lisans eğitimine başladı. 2009 yılında Milli Eğitim Bakanlığına bağlı devlet kurumlarında bilgisayar öğretmeni olarak görev yapmaktadır.