MADEN HAVALANDIRMA
ŞEBEKELERİNİN OPTİMİZASYONU

M. Saim SARAÇ

Anadolu Üniversitesi
Fen Bilimleri Enstitüsü
Lisansüstü Yönetmeliği Uyarınca
Maden Mühendisliği Anabilim Dalında
DOKTORA TEZİ
Olarak Hazırlanmıştır.

Danışman : Doç.Dr. M. Rifat BOZKURT

Nisan-1988

M. Saim SARAÇ'ın DOKTORA tezi olarak hazırladığı "MADEN HAVALANDIRMA ŞEBEKELERİNİN OPTİMİZASYONU" başlıklı bu çalışma, jürimizce lisansüstü yönetmeliğinin ilgili maddeleri uyarınca değerlendirilerek kabul edilmiştir.

..../..../....

Başkan : Doç.Dr.Erdil Ayvazoğlu

Üye : Doç.Dr. Erkin NASUF

Üye : Doç.Dr. Rifat BOZKURT

Fen Bilimleri Enstitüsü Yönetim Kurulu'nun .6.6.1988.. gün ve ...178-1..... sayılı kararıyla onaylanmıştır.

Enstitü Müdürü

Prof.Dr. Rüstem KAYA

ÖZET

Bu çalışmada, havalandırma şebekelerinin kısıtsız optimizasyon teknikleri ile çözümlenebilirliği incelenmiş ve havalandırma şebeke hesaplamalarında maden mühendisine getirebileceği kolaylıklar araştırılmıştır.

Havalandırma şebekeleri için çok değişkenli, doğrusal olmayan, kısıtsız bir model oluşturulmuştur. Model, şebekede tüketilen hava gücünün üçte birini ifade etmektedir. Bu durumda problem, şebekedeki güç tüketimini enküçükleyen hava miktarı değerlerinin hesaplanması olarak şekillenmektedir.

Modelin çözümünde, amaç fonksiyonunun türevini değerlendiren kısıtsız optimizasyon teknikleri kullanılmıştır. Bu tekniklerden En Hızlı İniş, Fletcher-Reeves, Newton ve Davidon-Fletcher-Powell yöntemleri için bilgisayar programları yazılmıştır. Örnek şebekeler üzerinde uygulamalar yapılarak, havalandırma şebekelerinin kısıtsız optimizasyon teknikleri ile çözümlenebileceği kanıtlanmıştır.

Başlangıç hava miktarı değerlerinin sıfır olarak atanmasının kısıtsız optimizasyon tekniklerinin çalışmasında önemli bir olumsuz etki yapmadığı belirlenmiştir. Davidon-Fletcher-Powell yönteminin göz seçimine büyük kolaylıklar getirdiği saptanmıştır. Kısıtsız optimizasyon teknikleri sonuca daha az iterasyonda ulaşabilmekle birlikte, çözüm süresi bakımından Hardy Cross yöntemine üstünlük sağlayamamıştır.

Kısıtsız optimizasyon tekniklerinin ilk bir kaç iterasyonda optimum noktaya oldukça yaklaşabildikleri, daha sonraki iterasyonlarda ise yavaş bir yakınsama gösterdikleri belirlenmiştir. Hardy Cross yöntemi ise tersine bir yaklaşım göstermiştir. Her iki avantajı birleştiren bir kombine yöntem geliştirilerek, örnek şebekelere uygulanmıştır.

Kombine yöntemin gerek iterasyon sayısı, gerekse çözüm süresi bakımından Hardy Cross tekniğine oranla daha avantajlı olduğu belirlenmiştir. Bu yöntem için bir bilgisayar programı yazılarak, uygulamacı maden mühendisinin kullanımına sunulmuştur.

Anahtar Kelimeler : Maden Havalandırma, Kısıtsız Optimizasyon.

SUMMARY

In this study, the solvability of mine ventilation networks by unconstrained optimization techniques has been analysed and their probable advantages have been investigated.

A multi-dimensional, nonlinear and unconstrained model for mine ventilation networks has been formed. The model represents one third of the air power consumed in the network. In this case, the problem appears as the power consumption in the network.

In the solution of the model, unconstrained optimization techniques which evaluate the derivation of the objective function have been used. Of these techniques, computer programmes have been prepared for Steepest Descent, Fletcher-Reeves, Newton and Davidon-Fletcher-Powell methods. The solvability of ventilation networks by unconstrained optimization techniques was proved, by the applications on network models.

It has been determined that, the assignation of initial quantity values as zero, doesn't create any important negative effect on the performance of unconstrained optimization techniques. It was also found out that Davidon-Fletcher-Powell method provides great facilities with mesh selection. Although unconstrained optimization techniques might reach to the solution by less iteration, with respect to the solution time it's not superior over Hardy Cross method.

It has been determined that, unconstrained optimization techniques could nearly approach to the optimum point within first few iterations, and show a slower convergence speed in the later iterations. It was found out that Hardy Cross method has an opposite trend. A combined method which includes both advantages has been developed and applied to the network models.

It has been concluded that Combined method is advantageous compared with Hardy Cross method, both from the point of iteration number and solution period. A computer programme of the method has been prepared and presented to the use of Mining Engineers.

Keywords : Mine Ventilation, Unconstrained Optimization.

TEŞEKKÜR

Bu tezin yürütülmesinde danışmanlığımı yaparak görüş ve önerileriyle bana yol gösteren değerli hocam Doç.Dr. M.Rifat BOZKURT'a, lisans öğrenimimden başlayarak tüm yüksek öğrenim yaşamımda bilgilendiğim, doktora tezimin her aşamasında beni yönlendiren İ.T.Ü. Maden Fakültesi Öğretim Üyesi Doç.Dr. Erdil AYVAZOĞLU'na, Şebeke Analizi ve Optimizasyon konularında fikirlerinden yararlandığım Ege Üniversitesi Öğretim Üyesi Doç. Orhan YÜKSEL'e, çalışmalarım süresince desteklerini gördüğüm Anadolu Üniversitesi, Mühendislik-Mimarlık Fakültesi yetkilileri ile Maden Mühendisliği Bölümündeki çalışma arkadaşlarıma, İ.T.Ü. Maden Fakültesi Öğretim Üyelerine, tüm yaşamım boyunca büyük maddi ve manevi fedakarlıklara katlanarak eğitimimi sağlayan aileme teşekkürlerimi sunarım.

İÇİNDEKİLER

İÇİNDEKİLER (Devam ediyor)

İÇİNDEKİLER (Devam ediyor)

Sayfa

İÇİNDEKİLER (Devam ediyor)

İÇİNDEKİLER (Devam ediyor)

ŞEKİLLER DİZİNİ

ŞEKİLLER DİZİNİ (Devam ediyor)

ŞEKİLLER DİZİNİ (Devam ediyor)

ÇİZELGELER DİZİNİ

ÇİZELGELER DİZİNİ (Devam ediyor)

SİMGELER VE KISALTMALAR DİZİNİ

Simgeler	Açıklama
D	Şebekenin göz matrisi.
d_{ij}	Göz matrisinin elemanları.
E	Şebekenin durum matrisi.
e_{ij}	Durum matrisinin elemanları.
F_i	Fibonacci katsayıları.
H	Hessien matrisi.
H^{-1}	Hessien matrisinin inversi.
k'	Altın Oran.
P	İlerleme yönü vektörü.
r	Optimal adım boyutu.
S_q	Tahminin standard hatası.
Q°	Başlangıç Q değerleri seti, m^3/s.
U	Amaç fonksiyonu.
∇U	Amaç fonksiyonunun gradyan vektörü.

Kısaltmalar	Açıklama
E.H.İ.	En Hızlı İniş Yöntemi.
F-R	Fletcher-Reeves Yöntemi.
D-F-P	Davidon-Fletcher-Powell Yöntemi.
N.	Newton Yöntemi.
H.C.	Hardy Cross Yöntemi.
S.H.	Tahminin Standard Hatası.
İ.S.	İterasyon Sayısı.
Ç.S.	Çözüm Süresi.

1. GİRİŞ

Yeraltı ocaklarının havalandırılması maden mühendisliği disiplini için gerek planlama aşamasında, gerekse günlük çalışmalarda önemli bir uğraşı alanıdır.

Havalandırma çalışmalarının amacı kısaca:

a) Ocaktaki canlıların hava gereksinimini karşılamak,

b) Tehlikeli gaz birikimlerini etkisiz kılmak,

c) Optimum bir ocak iklimi elde etmek,

d) Açık alevli lambalar ve patlarlı motorlar için oksijen gereksinimini karşılamak,

e) Tozlu havayı fazla miktarda hava ile seyreltmek veya zararsız hale getirmek,

olarak özetlenebilir (Saltoğlu, 1975).

Maden mühendisinin karşılaşacağı havalandırma sorunları ise, yeni damarlarda üretime geçmek, var olan bir veya birkaç vantilatörü şebekeden çıkarmak, devreye yeni vantilatörler ilave etmek, yeni galerilerin sürülmesi veya bazı eski galerilerin kapatılması, doğal havalandırmanın ve ocak yangınlarının etkisinin analiz edilmesi olarak şekillenebilir (Ayvazoğlu, 1986).

Ayvazoğlu (1973), yeraltı madencilik çalışmalarında havalandırma maliyetinin, genel maliyetin ancak %1 kadarını oluşturmasına karşılık, uygun olmayan bir havalandırma sisteminin üretim çalışmalarında çok büyük olumsuzluklar yaratacağını belirtmektedir.

Havalandırmadan kaynaklanan sorunlardan dolayı malzeme ve ekipman kaybına neden olunduğu, üretim yapılan panoların kapatılarak yerine bir daha konulamayacak ulusal servetin yeraltında terk edilmek zorunda kalındığı bir gerçektir.

Getirdiği maddi kayıplardan çok daha önemli olarak, iyi projelendirilmeyen ve peryodik denetimleri titiz bir şekilde yapılmayan havalandırma sistemlerinin, toplu can kayıplarına yol açtığı dünya ve maalesef özellikle ülkemiz madencilik pratiğinde iyi bilinmektedir.

Bir havalandırma sisteminin projelendirilmesinde kol direnç değerleri bilinen, bir veya birkaç vantilatör içeren şebekenin kollarındaki hava dağılımının hesaplanması istenir. Bu tip şebekeler "Doğal Dağılımlı Şebekeler" olarak nitelenir (Hartman, 1982).

Çizgisel diyagramları paralel ve seri bağlı devreler haline getirilebilen basit şebekeler için kollardaki hava dağılımının hesaplanması, elektrik devrelerine benzer şekilde Eşdeğer Dirençler yöntemi ile kolayca yapılabilir.

Ancak pratikte havalandırma şebekelerinin büyük çoğunluğu, paralel ve seri bağlı devrelere dönüştürülemeyen kompleks şebekeler durumundadır. Bu tür şebekelerin çözümlenmesi ise yoğun ve karmaşık matematiksel işlemleri gerektirmekte, hesaplamalarda sayısal bilgisayarların kullanılması kaçınılmaz olmaktadır.

Tarihi gelişim içinde, kompleks şebekelerin analizi amacıyla çeşitli analog ve sayısal yöntemler kullanılmakla birlikte, sonraki yıllarda Hardy Cross yaklaşık tekrarlama (iterasyon) tekniği en uygun metod olarak kabul görmüştür (Güyagüler, 1982). Literatürde bu tekniği temel alan ve ayrıntılarda farklılıklar gösteren pek çok bilgisayar programı bulunmaktadır.

Bazı araştırmacılar tarafından son yıllarda yapılan yeni çalışmalarda, probleme değişik matematiksel yaklaşımlar da geliştirilmiştir. Wang (1984) ile Ueng ve Wang (1984), "Bileşke Yön" tekniğinin bu amaçla kullanılabileceğini ileri sürmüşlerdir. Bhamidipati ve Procarione (1985; 1986), kollardaki hava dağılımının "Parçalı Doğrusal Yaklaşım" (Piece Linearization) tekniği ile hesaplanabileceğini,

ancak uygulamada Hardy Cross yöntemine üstünlük sağlanamadığını rapor etmektedirler.

Yeraltı maden işletmeciliğinde yaşamsal bir öneme sahip olan havalandırma sistemlerinin projelendirilmesine yeni yaklaşımlar getirmek, bilim adamları için daima ilgi çekici bir araştırma konusu olmuştur. Matematik programlama bilim dalındaki ve bilgisayar teknolojisindeki gelişmeler, yeni hesaplama tekniklerinin mühendislik problemlerine uygulanmasına olanak tanımıştır.

Bu tezde, kompleks havalandırma şebekelerinin analizinde klasik Hardy Cross yöntemine oranla daha hızlı ve daha kullanışlı bir çözüm tekniğinin araştırılması ve havalandırma mühendisine hesaplama kolaylıkları getirilmesi amaçlanmıştır. Bu amaç doğrultusunda kısıtsız optimizasyon tekniklerinin havalandırma şebeke hesaplamalarında kullanılabilirliği araştırılmış, ele alınan teknikler ve Hardy Cross tekniğinin çalışma performansları karşılaştırılarak test edilmiştir.

2. HAVALANDIRMA ŞEBEKE HESAPLARI

2.1. Giriş

Havalandırma şebekelerinin projelendirilmesinde öncelikle sistemin planı üzerinde çalışılarak kol ve kavşaklar sistematik şekilde numaralandırılır, sistemin çizgisel diyagramı çizilir ve vantilatörler işaretlenir.

Ocaktaki hava yollarının direnç değerleri ölçme veya hesaplama yoluyla belirlendikten sonra, diyagramdaki kolların eşdeğer dirençleri hesaplanır. Şebekedeki vantilatörler sabit basınç kaynağı olarak veya karakteristik eğrilerinin katsayıları belirlenerek hesaba katılır. Doğal hava akımının etkisi incelenerek hava yoğunluklarına göre değişimi belirlenir.

Havanın ocak yollarındaki akışında akışkanlar mekaniği yasaları geçerlidir. Ocak havası sıkışabilir bir akışkan olmakla birlikte, havalandırma şebeke analizinde ocak havasının sıkışmaz olduğu, termodinamik olarak bir hacım değişikliğine uğramadığı kabul edilir. Bu kabul hesaplamalarda büyük kolaylıklar sağlar ve hassasiyeti fazla etkilemez (Skochinsky and Komarov, 1969).

2.2. Basınç Düşüşü

Yeraltı yollarında hava yüksek basınçlı noktadan alçak basınçlı noktaya doğru hareket eder. Ocak içinde havanın bir noktadan başka bir noktaya hareketi sırasında, akışa karşı dirençlerden kaynaklanan belli bir basınç düşüşü oluşur.

Toplam basınç düşüşü, sürtünme kaybı ile yol üzerindeki engellerden kaynaklanan şok kaybının toplamı olarak ifade edilir. Gözönüne

alınan kolda vantilatör bulunuyorsa vantilatör basıncı basınç yaratıcı, doğal hava basıncı ise basınç yaratıcı veya basınç düşürücü bir etkileme yapar.

$$h = h_s + h_x \mp h_d - h_v \qquad (2\text{-}1)$$

h : Toplam basınç düşüşü (mm ss),

h_s : Sürtünme kaybı (mm ss),

h_x : Şok kaybı (mm ss),

h_d : Doğal hava basıncı (mm ss),

h_v : Vantilatör basıncı (mm ss).

Sürtünme kaybı galeri duvarlarına sürtünmeden dolayı oluşan kayıplardır ve basınç düşüşünün ana kaynağıdır. Sürtünme kaybı ile hava hızı arasında üssel bir ilişki olduğu bilinmektedir (Hartman, 1982).

$$h_s \propto V^n \qquad (2\text{-}2)$$

Buradaki n üssü akış türüne bağlı olarak 1,7-2,3 arasında değişmekte, Reynolds sayısı 20000 veya daha fazla olduğunda 2 değerini almaktadır. Bu sayıya karşılık gelen hız değeri ise 0,1 m/s civarındadır. Ocak havalandırmasında hava hızının 0,1 m/s'den daha az olması olasılığı çok zayıf olduğundan, havalandırma hesaplamalarında n değerini 2 olarak almak fazla bir hata getirmemektedir (Hall, 1981; Güyagüler, 1982).

Dairesel bir boruda akan akışkanın basınç düşüşü için Darcy tarafından verilen eşitlik üzerinde çalışmalar yapan Atkinson (1854), galerilerdeki sürtünme kaybı için,

$$h_s = \frac{k.W'.L.V^2}{F} \quad , \quad (\text{mm ss}) \qquad (2\text{-}3)$$

veya $V = Q/F$ dönüşümüyle,

$$h_s = \frac{k.W'.L.Q^2}{F^3} \quad , \quad (\text{mm ss}) \qquad (2\text{-}4)$$

eşitliklerini vermektedir (Scothinsky and Komarov, 1969'dan). Burada;

k : Sürtünme katsayısı ($kg.s^2/m^4$),

W' : Galeri çevresi (m),

L : Galeri uzunluğu (m),

F : Galeri kesiti (m^2),

V : Hava hızı (m/s),

Q : Hava miktarı (m^3/s)'dir.

(2-4) eşitliğinde hava miktarı dışındaki parametreler hava yolu ile ilgili büyüklüklerdir ve ancak hava yolu karakteristiklerinin değiştirilmesi durumunda değişirler. Bu sabit değerlerin "Kol Direnci" olarak nitelenen tek bir terimle,

$$R = \frac{k.W'.L}{F^3} \quad , \quad (\text{Kilomurg}) \qquad (2\text{-}5)$$

olarak ifade edilmesi durumunda basınç düşüşü eşitliği,

$$h_s = R.Q^2 \quad , \quad (\text{mm ss}) \qquad (2\text{-}6)$$

olarak basitleşmektedir.

Havalandırma şebeke hesaplamalarında (2-6) bağıntısının, şebekenin tüm kolları için geçerli olduğu kabul edilir (Hall,et.al., 1982).

2.3. Kirchoff Yasaları

Temelde elektrik şebekeleri için verilen Kirchoff yasaları havalandırma şebekeleri için de geçerlidir. Kirchoff I yasası olarak bilinen akımın korunumu ilkesine göre, bir kavşak noktasına gelen ve giden hava miktarlarının toplamı sıfırdır.

$$\sum_{i=1}^{n} Q_i = 0 \qquad (2\text{-}7)$$

Kirchoff II yasası olarak isimlendirilen basınç düşüşlerinin korunumu ilkesine göre de, kapalı bir göz etrafındaki basınç düşüşlerinin toplamı sıfırdır.

$$\sum_{j=1}^{m} R_j . Q_j^2 - \sum_{k=1}^{V} h_{v_k} = 0 \qquad (2\text{-}8)$$

2.4. Hardy Cross İterasyon Tekniği

2.4.1. Düzeltme Değeri

Kompleks havalandırma şebekelerinin analizinde kullanılan analitik, analog ve sayısal çözüm tekniklerinden en yaygın olanı Hardy Cross iterasyon yöntemidir.

Anılan teknik esas olarak şehir suyu dağılım ağı hesaplamaları için Hardy Cross (1936) tarafından ortaya konmuş, Scott-Hinsley (1951; 1952) ve McPherson'un (1966) çalışmaları ile havalandırma şebekelerine uyarlanarak geliştirilmiştir (Hall, et al., 1982'den).

Ocak yollarındaki hava akışı (2-6) eşitliğine uymaktadır. Bu ifadede h ve R değerleri bilindiğinde, üçüncü parametre olan Q hava miktarının gerçek değerinin saptanması ile havalandırma şebekesinin analizi sözkonusudur.

Bu analizde önce şebekenin her kolu için akımın korunumu ilkesini gerçeklemek koşuluyla, başlangıç Q_a değerleri rassal olarak atanır. Bu durumda gerçek Q değeri ile atanan Q_a değeri arasında ΔQ kadar hata olacaktır.

$$Q = Q_a + \Delta Q \qquad (2\text{-}9)$$

Böylece basınç düşüşü bağıntısı,

$$h = R.(Q_a + \Delta Q)^2 \qquad (2\text{-}10)$$

biçimine gelir. Bu eşitliğin açınımı,

$$h = R.(Q_a^2 + 2.Q_a.\Delta Q + \Delta Q^2) \qquad (2\text{-}11)$$

olarak şekillenir. İkinci dereceden terimi ihmal ederek bulunan,

$$h = R.Q_a^2 + 2.R.Q_a.\Delta Q \qquad (2\text{-}12)$$

ifadesinden ΔQ çekilirse,

$$\Delta Q = \frac{h - R.Q_a^2}{2.R.Q_a} \qquad (2\text{-}13)$$

düzeltme değerini veren eşitlik elde edilir. Bu ifadeden hesaplanan düzeltme miktarı Q_a başlangıç değerine uygulanır. Düzeltme işlemleri tekrarlı olarak bir çok kez yapılırsa, gerçek Q değerine istenen hassasiyette yaklaşılır.

$$Q_a \tilde{=} Q \qquad (2\text{-}14)$$

Tek bir kol için verilen hesaplama mantığı bir şebekeye uyarlandığında, bu şebekeden seçilen kapalı gözler için düzeltme değerleri hesaplanır. Bir göz etrafındaki basınç düşüşlerinin toplamı sıfır olduğundan, m'inci göze uygulanacak düzeltme için,

$$\Delta Q_m = - \frac{\Sigma(R.Q_a^2) - h_v}{\Sigma(2.R.Q_a)} \qquad (2\text{-}15)$$

eşitliği elde edilir. Bu eşitliğin payı m'inci göz etrafındaki toplam basınç düşüşünün ifadesi olup, hava akış yönüne göre pozitif veya negatif olması sözkonusudur. Payda ise kol karakteristik eğrilerinin eğimlerinin toplamıdır ve daima pozitif işaretlidir. Bu işaret siste-

mini gerçekleyerek ve vantilatör karakteristik eğrisinin eğimi (S) ile gözdeki doğal havalandırma basıncını da hesaba katarak,

$$\Delta Q_m = - \frac{\Sigma(R.|Q_a|.Q_a) - h_v - h_d}{\Sigma(2.R.|Q_a|) - S} \qquad (2\text{-}16)$$

eşitliğine ulaşılır.

Elde edilen ifadenin şebekeden seçilen gözlere tekrarlı olarak uygulanması ile şebekenin her kolundaki gerçek hava miktarı belli bir yaklaşıklıkla hesaplanmış olur.

2.4.2. Göz seçimi

Düzeltme değerlerinin şebekedeki olası her göz için hesaplanması zorunlu olmayıp, kol sayısı N, kavşak sayısı K kadar olan bir şebeke için,

$$G = N - K + 1 \qquad (2\text{-}17)$$

sayıda göz seçerek, düzeltme işlemlerinin bu gözler için yapılması yeterli olmaktadır. Ancak tüm kollardaki hava miktarlarını hesaplayabilmek bakımından her kol en azından bir kez ele alınmalıdır.

Büyük dirençli kollar hesaplamalarda iterasyon sayısını ve çözüm süresini artırır. Bu nedenle büyük dirençli kolların birden fazla gözde yer almamasına dikkat edilir. Bunu sağlamak için göz sayısı kadar büyük dirençli kol "Ana Kol" olarak ayrılır, diğer kollar ise "Tali Kollar" olarak nitelenir.

Her gözün ilk kolu ana kollardan birisi olmalı, gözdeki tali kollar kapalı bir göz oluşturmamalıdır. Her gözde bir tek ana kol bulunmalı, her ana kol sadece bir gözde yer almalıdır. Saat akrebinin dönüş veya ters yönü referans yön olarak kabul edildikten sonra, göz di-

zileri oluşturulurken ele alınan kol numarasına, hava akış yönü referans yönle aynı ise (+), farklı ise (-) işaret verilmelidir.

2.4.3. Yöntemin algoritması

Hardy Cross tekniğinin algoritması McPherson (1966) tarafından aşağıdaki gibi verilmektedir:

1. Adım : Akımın korunumu ilkesini sağlayacak şekilde, her kol için başlangıç Q° değerleri ve istenen hassasiyet değeri (ε) atanır, vantilatör basıncı belirlenir.

2. Adım : Göz dizileri oluşturulur.

3. Adım : Her göze uygulanacak düzeltme değeri (2-16) eşitliğinden hesaplanır ve hesaplanan değer gözün kollarındaki hava miktarlarına uygulanır.

4. Adım : $|\Delta Q_{max}| < \varepsilon$ ise işlem bitirilir, tersi durumda üçüncü adıma dönülür.

2.4.4. Bilgisayar programı

Literatürde Hardy Cross tekniğini temel alan, gözleri otomatik olarak seçip seçmemeye, doğal havalandırmanın, ocak yangınlarının, hacım değişmesinin etkisini analiz edip etmemeye göre farklılıklar gösteren pek çok bilgisayar programı bulunmaktadır. Havalandırma şebekelerinin analizi için Ayvazoğlu (1973) tarafından FORTRAN II dilinde verilen bilgisayar programı BASIC programlama diline çevrilip, bazı katkılar da yapılarak bu çalışmada kullanılmıştır.

Göz dizileri elle seçildikten sonra programa veri olarak okutulmakta, doğal havalandırmanın etkisi analize sokulmamaktadır. Vantilatör tarafından yaratılan basınç farkı, vantilatör karakteristik eğrisinin katsayılarının veri olarak okutulması ile, vantilatörün bulunduğu

kol akımı için,

$$h_v = A.Q^2 + B.Q + C \qquad (2\text{-}18)$$

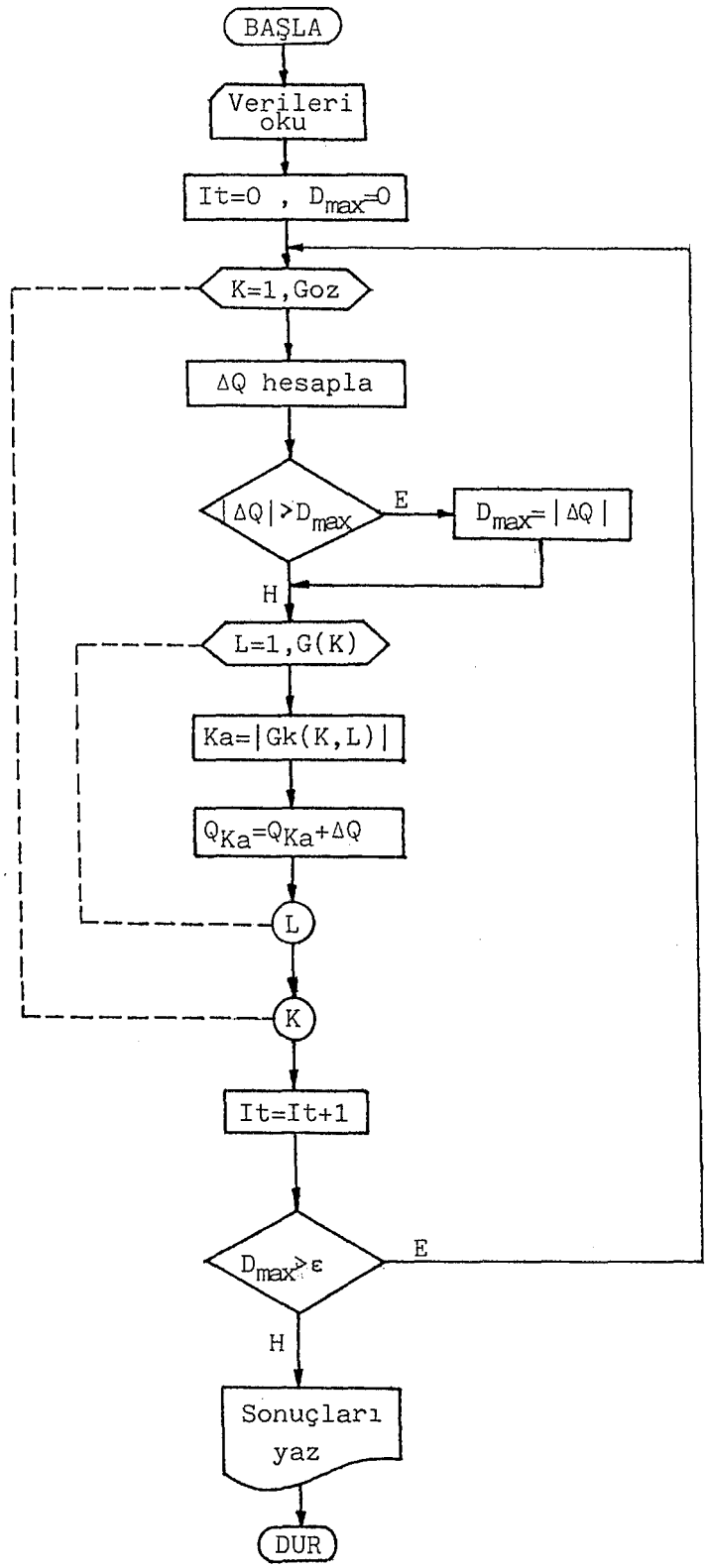
polinomundan hesaplanmaktadır. Burada,

A ; B ; C : Vantilatör karakteristik eğrisinin katsayıları,

Q : Vantilatörün bulunduğu koldaki hava miktarıdır.

İşlemlere her iterasyonda hesaplanan maksimum düzeltme değeri, hassasiyet derecesinin altına düşene kadar devam edilmektedir. İstenen hassasiyet sağlandığında döngü dışına çıkılarak, şebekenin her kolundaki hava miktarı ve basınç düşüşü değerleri tablo düzeninde basılmaktadır. Hassasiyet değeri uygulamalarda ε=0,005 m^3/s olarak kullanılmıştır.

Eklerde verilen örnek şebekelerin Hardy Cross tekniği ile çözümlenmesinde kullanılan bilgisayar programı EK-5'de, genelleştirilmiş akış şeması ise Şekil 2,1'de verilmektedir.



Şekil 2.1 : Hardy Cross tekniği bilgisayar programının genelleştirilmiş akış şeması.

3. DOĞRUSAL OLMAYAN PROGRAMLAMADA BAZI KAVRAM VE ÖZELLİKLER

3.1. Giriş

Doğrusal olmayan programlama tekniklerini içeren Yöneylem Araştırması bilim dalı, İkinci Dünya Savaşı sırasındaki hava akınları karşısında en iyi savunma şeklini belirlemede, radarların etkin bir şekilde kullanılmasını sağlamak amacıyla, farklı disiplinlerden bilim adamlarının çalışmaları ile doğmuş, başarılı sonuçlar alınmasından sonra diğer sanayi problemlerine uygulanması ile büyük ilerlemeler göstermiştir.

Yöneylem Araştırması örgütlerin problemlerini sistemin bütünü içinde, disiplinlerarası ekiple ve bilimsel yöntem uygulamasıyla belirlemek ve bunlara çözüm bulmak şeklinde tanımlanır (Kara, 1985).

Yöneylem Araştırmasının problemlere bilimsel yaklaşımı:

- Problemin belirlenmesi,
- Matematiksel modelin geliştirilmesi,
- Modelin çözülebilirliğinin sağlanması,
- Modelin çözümü ve kanıtlanması evrelerinden oluşur.

"Matematiksel Model", sistemin mantıksal akış modelinin sembollerle gösterimidir. Sistemin verilen koşullarda amacına uygun eniyi davranışı gösterebilmesi için uygulanacak eylemleri belirleyen matematiksel modeline "Karar Modeli" denir. Bir karar modeli sistemin amaçlarına ulaşabilmesi için karar değişkenlerine verilmesi gereken değerleri belirler. Bir karar modelinin temel bileşenleri:

- Karar değişkenleri,
- Parametreler,

- Kısıtlar,
- Amaç fonksiyonu, olarak sıralanabilir.

Sistemin davranışını etkileyen ve alabileceği değerler karar verici tarafından saptanan bileşenlere "Karar Değişkeni", değerlerinde karar vericinin hiç bir etkisi olmayan bileşenlere de "Parametre" denir. Karar değişkenlerinde ve karar değişkenleri ile parametreler arasında gerçekleşmesi gereken zorunlu ilişkiler "Kısıtlar" olarak modele katılır. "Amaç Fonksiyonu" ise eniyilenmesi istenen ilişkinin matematiksel ifadesidir.

Sistemin istenen davranışa getirilebilmesi için karar değişkenlerinin alacağı değerleri araştırma çalışmaları "Programlama" kavramı ile ifade edilir. Programlama teknikleri modelin yapısına göre beş başlık altında toplanabilir (Kara, 1985):

- Doğrusal Programlama,
- Doğrusal olmayan programlama,
- Tamsayılı programlama,
- Dinamik programlama,
- Stokastik (Rassal) programlama.

3.2. Tanım ve Genel Yapı

Bir problem için geliştirilen karar modelinin amaç fonksiyonunun veya kısıtlarından birinin doğrusal olmaması durumunda, kullanılan kavram ve teknikler "Doğrusal Olmayan Programlama" olarak tanımlanmaktadır (Kara, 1986).

Doğrusal olmayan karar modelinin genel yapısı, amaç fonksiyonu veya kısıtlardan en az birinin doğrusal olmaması koşuluyla;

Amaç fonksiyonu :

$$U_{\substack{max \\ min}} = f(x_1, x_2, \ldots, x_n)$$

Kısıtlar :

$$g_i(x_1,x_2,\ldots,x_n) \leqslant b_i \quad , \quad (i=1,2,\ldots,m)$$
$$x_1,x_2,\ldots,x_n \geqslant 0$$

olarak verilir (Tulunay, 1980). Kısıtlar eşitlik ve/veya eşitsizlik halinde olabilir.

Matematiksel modele göre, amaç denklemini eniyileyen ve kısıtlayıcı şartları gerçekleyen bir $(x_1,x_2,\ldots,x_n)$ seti problemin optimum çözümünü verir.

Verilen genel matematiksel forma uyan her problemi çözebilecek, genel bir doğrusal olmayan programlama tekniği yoktur. Çözülecek problemin amaç fonksiyonu ve kısıtlarının yapılarına göre, doğrusal olmayan programlama çözüm tekniklerinden en uygun olanı ile çözüme gidilir.

3.3. Konvekslik-Konkavlık

Tanımlı olduğu aralıkta bir fonksiyonun eğrisi üzerinde alınan herhangi iki noktayı birleştiren doğru parçası fonksiyon eğrisinin üzerinde kalırsa "Konveks", fonksiyon eğrisinin altında kalırsa da "Konkav" bir fonksiyondan söz edilir (Hillier and Lieberman, 1974).

Tek değişkenli bir f(x) fonksiyonunun eğrisi üzerinde alınan herhangi iki P ve R noktalarını birleştiren $\overline{PR}$ doğru parçası üzerinde gözönüne alınan herhangi bir M noktasının koordinatları;

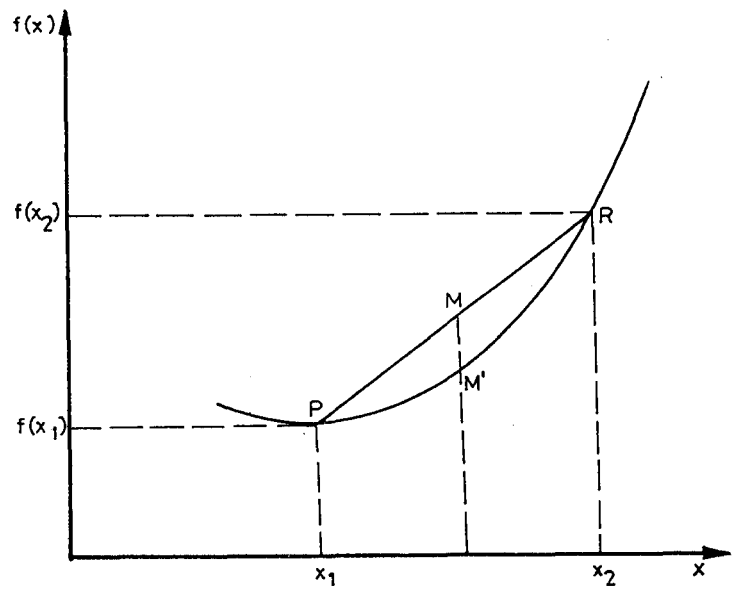
$$M[x_1+\lambda.(x_2-x_1) \; ; \; f(x_1)+\lambda.(f(x_2)-f(x_1))] \qquad (3\text{-}1)$$

olarak, M noktasının fonksiyon eğrisi üzerindeki izdüşümü olan M' noktasının koordinatları ise;

$$M'[x_1+\lambda.(x_2-x_1) \; ; \; f(x_1)+\lambda.(x_2-x_1)] \qquad (3\text{-}2)$$

olarak ifade edilebilir.

M ve M' noktalarının apsisleri aynı olduğundan $0 \leq \lambda < 1$ aralığındaki her nokta için M noktasının ordinatının M' noktasının ordinatından büyük olması durumunda, bir başka deyimle $\overline{PR}$ doğru parçasının daima fonksiyon eğrisi üzerinde kalması durumunda fonksiyon konveks olacaktır. Benzer şekilde, M' noktasının ordinatının M noktası ordinatından daima büyük olması durumunda ise, fonksiyon konkav bir fonksiyon olacaktır.



Şekil 3.1 : Konveks bir f(x) fonksiyonu eğrisi.

Matematiksel olarak bir f(x) fonksiyonu $0 \leq \lambda \leq 1$ aralığında olmak üzere, x'in herhangi iki x_1 ve x_2 değerleri için;

$$f[x_1+\lambda.(x_2-x_1)] \leq f(x_1)+\lambda.[f(x_2)-f(x_1)] \qquad (3\text{-}3)$$

koşulunu sağlıyorsa f(x) konveks bir fonksiyon olur (Bradley, et al., 1977). Burada eşitsizliğin sol tarafı gözönüne alınan noktada fonksiyonun değerini, sağ tarafı ise doğrusal interpolasyonu ifade etmektedir. $0 < \lambda < 1$ aralığında,

$$f[x_1+\lambda.(x_2-x_1)] < f(x_1)+\lambda.[f(x_2)-f(x_1)] \qquad (3\text{-}4)$$

koşulu sağlanıyorsa f(x) "Tam Konveks" bir fonksiyondur.

Tek değişkenli bir fonksiyonun konveks veya konkavlığını belirlemede, fonksiyonun ikinci türevinden de yararlanılabilir. (a-b) aralığında tanımlı olan bir f(x) fonksiyonu, bu aralıkta sürekli bir ikinci türeve sahipse;

a) (a-b) aralığındaki tüm x değerleri için;

   $f''(x) \geq 0$ ise bu aralıkta konveks,

   $f''(x) > 0$ ise tam konvekstir.

b) (a-b) aralığındaki tüm x değerleri için;

   $f''(x) \leq 0$ ise bu aralıkta konkav,

   $f''(x) < 0$ ise bu aralıkta tam konkavdır (Gaver and Thompson, 1973).

Tek değişkenli fonksiyonlar için verilen bu kurallar çok değişkenli fonksiyonlar için de geçerlidir. Burada x yerine $(x_1,x_2,..,x_n)$ seti, f(x) yerine de $f(x_1,x_2,..,x_n)$ fonksiyonu gelecektir. Böyle bir fonksiyonda x seti n boyutlu Öklit Uzayında bir nokta tanımlayacaktır.

m=n+1 olmak üzere $(x'_1,x'_2,..,x'_n)$ ve $(x''_1,x''_2,..,x''_m)$ gibi fonksiyon eğrisi üzerindeki iki noktayı birleştiren doğru parçası üzerinde herhangi bir noktanın koordinatları;

$$M[\lambda.x''_1+(1-\lambda).x'_1 \; ; \; ..... \; ; \; \lambda.x'_m+(1-\lambda).x'_m] \qquad (3-5)$$

olarak belirlenir (Hillier and Lieberman, 1974).

$f(x_1,x_2,..,x_n)$ fonksiyonunun eğrisi üzerinde alınan her nokta çiftlerini birleştiren doğru parçaları eğri üzerinde kalıyorsa fonksiyon konveks, bu doğru parçaları eğriye sadece uç noktalarda değiyorsa tam konveks bir fonksiyon olur (Koo, 1977).

Çok değişkenli fonksiyonların konveks veya konkavlığı Hessien matrisinin oluşturulması ile de analiz edilebilir.

Bir $f(x_1,x_2,..,x_n)$ fonksiyonu onun (nXn) boyutlu Hessien matrisi, tüm mümkün $(x_1,x_2,..,x_n)$ setleri için pozitif belirli olduğunda tam konveks, pozitif yarı belirli olduğunda konvekstir (Tulunay, 1980).

n değişkenli bir fonksiyonun Hessien matrisi;

$$H = \left| \frac{\partial^2 f}{\partial x_i . \partial x_j} \right| \qquad (3\text{-}6)$$

olarak ifade edilir (Hadley, 1964). Bu matrisin $M_1,M_2,..,M_n$ determinantları Hessien matrisinin asal minörleri olarak nitelenir.

$$M_n = \begin{vmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots\cdots & \frac{\partial^2 f}{\partial x_1 . \partial x_n} \\ \vdots & & \\ \frac{\partial^2 f}{\partial x_n . \partial x_1} & & \frac{\partial^2 f}{\partial x_n^2} \end{vmatrix} \qquad (3\text{-}7)$$

Hessien matrisinin cinsini belirlemek için asal minörlere bakılır;

a) $M_1 \leqslant 0, M_2 \geqslant 0, \ldots, M_n (-1)^n \geqslant 0$ ise matris negatif yarı belirli, fonksiyon konkav,

b) $M_1 < 0, M_2 > 0, \ldots, M_n (-1)^n > 0$ ise matris negatif belirli ve fonksiyon tam konkav,

c) $M_1 \geqslant 0, M_2 \geqslant 0, \ldots, M_n \geqslant 0$ ise matris pozitif yarı belirli ve fonksiyon konveks,

d) $M_1 > 0, M_2 > 0, \ldots, M_n > 0$ ise matris pozitif belirli ve fonksiyon tam konvekstir (Loomba and Turban, 1974).

Doğrusal olmayan programlama problemlerinde amaç fonksiyonu ve kısıt fonksiyonlarının tiplerinin araştırılması önemlidir. Fonksiyonun konveks veya konkav gibi özel tip bir fonksiyon olması, kullanılacak çözüm tekniğini belirler ve çözümde büyük kolaylıklar sağlar.

Fonksiyonların konveks veya konkavlığının analizinde kullanılan ve ispatları temel matematik kitaplarından bulunabilecek bazı temel teoremler aşağıda verilmiştir :

Teorem 1 : Konveks fonksiyonların toplamı yine bir konveks fonksiyon, konkav fonksiyonların toplamı da konkav bir fonksiyon oluşturur.

Teorem 2 : Konveks fonksiyonların toplamında fonksiyonların en azından bir tanesi tam konveks ise tam konveks bir fonksiyon, aynı şekilde konkav fonksiyonların toplamında, en azından bir eleman tam konkav ise, tam konkav bir fonksiyon sözkonusu olur (Kowalik and Osborne, 1968).

Teorem 3 : Bir $f(x_1,x_2,..,x_n)$ fonksiyonu konveks ise,
$g(x_1,x_2,..,x_n)=-f(x_1,x_2,..,x_n)$
ters fonksiyonu konkav, $f(x_1,x_2,..,x_n)$ konkav ise, ters fonksiyonu konveks bir fonksiyon gösterir.

3.4. Süreklilik

Bir f(x) fonksiyonu (a-b) aralığında tarifli ve bu aralıktaki herhangi bir nokta v olsun. Fonksiyonun v ve v'nin çok küçük bir t miktarı kadar arttığı (v+t) noktaları arasındaki artma miktarı,

$$f(v+t) - f(v) = k \qquad (3\text{-}8)$$

olur. Değişkenin t artmasının ve fonksiyonun k artmasının aynı zamanda sıfıra yaklaşmaları durumunda, başka bir deyişle,

$$\lim_{t\to o} [f(v+t) - f(v)] = 0 \qquad (3\text{-}9)$$

olması durumunda f(x) fonksiyonunun x=v noktasında sürekli olabilmesi için;

a) Fonksiyonun x=v noktasında tanımlı olması,

b) $\lim_{x \to v} f(x)$ 'in var olması,

c) $\lim_{x \to v} f(x) = f(v)$ olması gerek ve yeterdir (Jeffrey, 1979).

Bir fonksiyon bir aralığın her noktasında sürekli ise, bu aralıkta sürekli demektir. Bu durumda fonksiyonun grafiği bu aralıkta kesiksiz bir eğri gösterir. Sürekli fonksiyonların cebirsel toplamı da sürekli bir fonksiyon oluşturur.

3.5. Optimum Çözüm

Doğrusal olmayan problem için geliştirilen matematiksel modelde kısıtları gerçekleyen herhangi bir X vektörü "Mümkün Çözüm", tüm modeli gerçekleyen $x^\circ=(x_1^\circ,\ldots,x_n^\circ)^T$ sütun vektörü "Optimal Nokta" olarak isimlendirilir. Optimum noktada amaç fonksiyonunun $f(x^\circ)$ değeri "Optimal Değer" veya "Optimum Değer" olarak nitelenir ve x° ile $f(x^\circ)$ değerleri "Optimum Çözüm"ü oluşturur (Phillips, et al., 1976).

Bir başka deyimle optimum çözüm, mümkün çözüm bölgesinde amaç fonksiyonunu eniyileyen noktadır (Simmons, 1975).

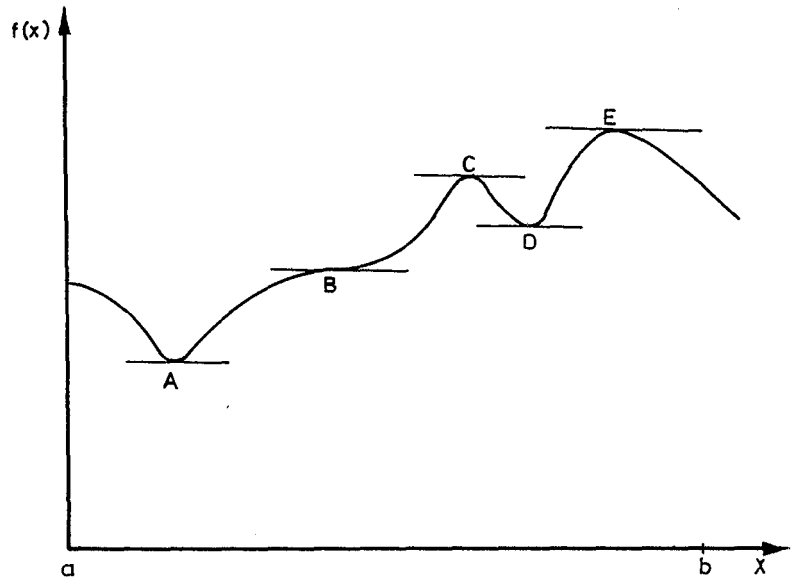
Doğrusal olmayan programlama tekniklerinde amaç, verilen problem için optimum çözüme ulaşmaktır. Ulaşılan çözümün optimum çözüm olabilmesi için gerek ve yeter şartları gerçeklemesi gerekir.

3.6. Ekstremum İçin Gerek ve Yeter Şartlar

3.6.1. Ekstremumlar

Bir fonksiyon birden çok maksimum veya minimum noktaya sahip olabilir. Eğrisi Şekil 3,2'deki gibi olan tek değişkenli bir f(x) fonksiyonunda birden çok ekstremum sözkonusudur.

Burada fonksiyon (a-b) aralığında A ve D noktalarında minimuma, C ve E noktalarında maksimuma, B noktasında da bir semer noktasına



Şekil 3.2 : Bir f(x) fonksiyonunda ekstremumlar.

sahiptir. E noktası bütünsel maksimumu, A noktası bütünsel minimumu göstermekte, diğer ekstremumlar ise yersel minimum ve yersel maksimum olarak nitelenmektedir (Tulunay, 1980).

Matematiksel olarak, çok değişkenli bir $f(x_1,x_2,..,x_n)$ fonksiyonu $X_o=(x_1,x_2,..,x_n)$ noktasında çok küçük bir t artışı için,

$$f(X_o+t) \leqslant f(X_o) \qquad (3\text{-}10)$$

oluyorsa, X_o noktasında bir maksimum vardır (Taha, 1976). Başka bir deyişle, fonksiyonun X_o'daki değeri, X_o noktasının komşuluğundaki değerlerinden daima büyük veya eşit oluyorsa X_o bir maksimum noktadır. Benzer şekilde,

$$f(X_o+t) \geqslant f(X_o) \qquad (3\text{-}11)$$

olması durumunda X_o bir minimum nokta gösterir.

Bir fonksiyonun eğimi fonksiyonun değerinin değişim oranını belirler. Eğimin sıfır olduğu noktalarda bir maksimum, bir minimum veya

bir semer noktası sözkonusudur. Bir başka deyimle, bu noktalardan fonksiyon eğrisine çizilen teğetler apsise paraleldir (Loomba and Turban, 1974).

3.6.2. Ekstremum için gerek şart

Verilen tanımlar doğrultusunda, tek değişkenli bir f(x) fonksiyonunun $x=x_o$ noktasında bir ekstremuma sahip olabilmesi için gerek şart,

$$f'(x) = 0 \quad , \quad (x=x_o\text{'da}) \tag{3-12}$$

olarak verilmektedir (Koo, 1977).

Çok değişkenli fonksiyonlar için de benzer şekilde $X=X_o$ setinin $f(x_1,x_2,..,x_n)$ fonksiyonunun bir ekstremumu olabilmesi için gerek şart,

$$\nabla f = \frac{\partial f(x_1,x_2,..,x_n)}{\partial x_j} = 0 \tag{3-13}$$

$(j = 1,2,...,n)$

olması, yani fonksiyonun değişkenlere göre kısmi türevlerinin sıfır olmasıdır (Beightler, et al., 1979).

3.6.3. Ekstremum için yeter şart

Tek değişkenli bir fonksiyonun x_o noktasında birinci türevinin sıfır olması şartı bu noktanın ekstremum olması için gerektir, ancak yeter değildir. Bir semer noktasında da eğim sıfır olmaktadır. Ayrıca bu nokta bir ekstremum olsa bile, minimum veya maksimumdan hangisi olduğu belirlenememektedir.

Tek değişkenli bir f(x) fonksiyonu $x=x_o$ noktasında gerek şartı sağlıyorsa, yeter şart için üç durum sözkonusu olabilir:

a) $f''(x_o) > 0$ ise x_o bir minimum noktadır,

b) $f''(x_o) < 0$ ise x_o bir maksimum noktadır,

c) $f''(x_o) = 0$ ise karar verebilmek için daha yüksek dereceli türevlere bakılır (Zahradnik, 1971).

n değişkenli bir fonksiyon X_o noktasında gerek şartı sağlıyorsa, yeter şart için Hessien matrisine bakılır:

a) H pozitif belirli ise X_o bir minimum noktadır,

b) H negatif belirli ise X_o bir maksimum noktadır,

c) H yarı belirli ise daha yüksek dereceli terimlere bakılır (Phillips, et al., 1976).

3.6.4. Konveks-Konkav fonksiyonlarda ekstremumlar

Fonksiyonun konveks veya konkav gibi özel tip bir fonksiyon olması durumunda ekstremumların araştırılması oldukça basitleşecektir. Bir fonksiyon x_o noktasında gerek şartı gerçekliyorsa, fonksiyonun konveks olması durumunda x_o noktası doğrudan doğruya bütünsel minimumu, fonksiyonun konkav olması durumunda ise bütünsel maksimumu gösterecektir. Bir başka deyişle (3-13) koşulu sadece gerek değil, yeter de olacaktır (Tulunay, 1980).

Bir konveks kümede tanımlı konveks bir fonksiyonun yersel minimum noktası aynı zamanda bütünsel minimum noktasıdır. Çünkü tek bir minimum nokta sözkonusudur. Benzer şekilde bir konkav kümede tanımlı konkav bir fonksiyonun yersel maksimum noktası aynı zamanda bütünsel maksimum noktasıdır (Bradley, et al., 1977 ; McCormick, 1983).

Bu özellik konveks fonksiyonların enküçüklenmesinde veya konkav fonksiyonların enbüyüklenmesinde büyük kolaylıklar sağlar. Fonksiyo-

nun konveks veya konkav olduğu belirlendikten sonra, gerek şart uyarınca (3-13) denklem sistemi oluşturulur. Bu sistemin köklerinin araştırılması ile sistemi sağlayan bir X_o seti bulunabilirse, bütünsel maksimum veya bütünsel minimum noktaya ulaşılabilecek, amaç fonksiyonunun eniyilenmesi gerçekleştirilmiş olacaktır.

3.7. Ayrılabilir Fonksiyon

n değişkenli bir fonksiyon,

$$f(x_1,x_2,...,x_n) = f(x_1) + f(x_2) + ... + f(x_n) \qquad (3\text{-}14)$$

şeklinde ifade edilebiliyorsa "Ayrılabilir Fonksiyon" olarak nitelenir (Gaver and Thompson, 1973). Bir başka deyimle, n değişkenli bir ayrılabilir fonksiyon tek değişkenli n tane fonksiyonun toplamıdır.

3.8. Kareli Biçimler

Çok değişkenli bir fonksiyon,

$$f(X) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}.x_i.x_j \qquad (3\text{-}15)$$

şeklinde ifade edilebiliyorsa fonksiyon bir "Kareli Fonksiyon" dur. Kareli fonksiyonun her teriminde ya iki değişkenin çarpımı veya bir değişkenin karesi vardır (Koo, 1977).

A matrisi (nXn) boyutlu bir kare matris ve X vektörü (nX1) boyutlu bir sütun vektörü olmak üzere, f(X) kareli fonksiyonu,

$$f(X) = \sum_{i=1}^{n} x_i \cdot \sum_{j=1}^{n} a_{ij}.x_j = X'.A.X \qquad (3\text{-}16)$$

şeklinde matris notasyonu ile ifade edilebilir (Hadley, 1964).

3.9. Tek Değişkenli Fonksiyonların Optimizasyonu

Belli bir aralıkta tek modlu olduğu belirlenen tek değişkenli fonksiyonların optimizasyonunda "Doğrusal Arama Yöntemleri" olarak nitelenen teknikler kullanılır.

Çok değişkenli optimizasyon tekniklerinde alt problem olarak bu yöntemlerin kullanılması gerekir ve doğrusal arama yapılması çözüm işlemlerinin en güç ve zaman alıcı kısmını oluşturur (Pierre, 1969). Bu nedenle çok değişkenli fonksiyonların eniyilenmesinde en uygun doğrusal arama yönteminin seçimi önemlidir.

Doğrusal arama yöntemleri aşağıdaki gibi sınıflanabilir:

I. Fonksiyonun türevini kullanmayan yöntemler:

1. Eşzamanlı Arama,
2. Ardışık Arama yöntemleri;
   a) Simetrik İki Nokta yöntemi,
   b) Altın Oran yöntemi,
   c) Fibonacci yöntemi,
   d) Kareli İnterpolasyon yöntemi.

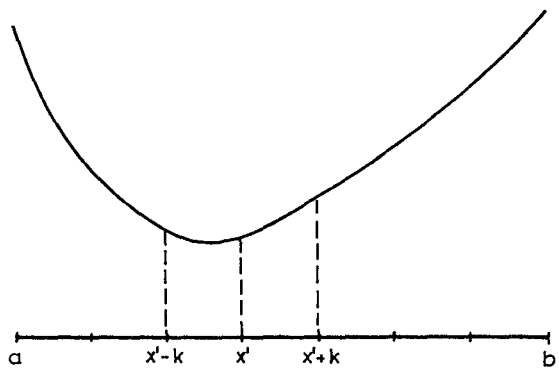
II. Fonksiyonun türevini kullanan yöntemler:

1. Newton yöntemi,
2. İkiye Bölme yöntemi,
3. Kübik Yaklaşım yöntemi.

3.9.1. Eşzamanlı Arama yöntemi

Minimum noktası aranan f(x) fonksiyonu için, x° minimum noktasının $a \leq x^\circ \leq b$ koşulunu gerçeklediği ve bu aralıkta fonksiyonun tek modlu olduğu biliniyorsa, (a-b) aralığı "Belirsizlik Aralığı" olarak tanımlanır (Wilde, 1964).

Eşzamanlı aramada belirsizlik aralığı, k uzunluğunda n adet alt aralığa bölünerek ızgara noktaları belirlenir ve fonksiyonun bu ızgara noktalarındaki değerleri hesaplanır. Fonksiyon en küçük değerini x' noktasında alıyorsa minimum x° noktası (x'-k ; x'+k) aralığında yer alacaktır (Sivazlian and Stanfel, 1975).



Şekil 3.3 : Eşzamanlı Arama yönteminde ızgaralama.

Bu aralık yeni belirsizlik aralığı olarak alınıp ızgaralama işlemi daha küçük bir k uzunluğu için tekrarlanır. Ardışık ızgaralama işlemleri ile minimum noktaya istenen hassasiyette yaklaşılır (Şekil 3,3).

3.9.2. Simetrik İki Nokta yöntemi

Bu yöntemde ızgaralama yerine, belirsizlik aralığı içinde simetrik iki nokta alınarak bu noktalarda fonksiyonun aldığı değerlere göre belirsizlik aralığı daraltılır. Belirsizlik aralığı istenen hassasiyete düşene dek, işleme ardışık olarak devam edilir (Şekil 3,4).

Yöntemin algoritması Bazaraa ve Shetty (1979) tarafından şu şekilde verilmektedir:

1. Adım : Yeterince küçük E>0 sabiti ve istenen hassasiyet değeri (ε) seçilir.

2. Adım : $(b-a)<\varepsilon$ ise 4. adıma gidilir,

Tersi durumda x_1 ve x_2 aşağıdaki ifadelerden hesaplanır.

$$x_1 = \frac{a+b}{2} - \frac{E}{2} \qquad (3\text{-}17)$$

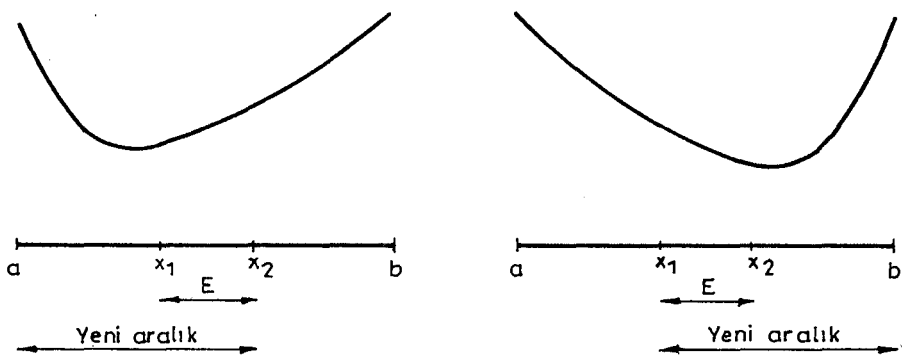
$$x_2 = \frac{a+b}{2} + \frac{E}{2} \qquad (3\text{-}18)$$

3. Adım : $f(x_1) < f(x_2)$ ise $a=a$, $b=x_2$

$f(x_1) \geq f(x_2)$ ise $a=x_1$, $b=b$

alınıp 1. adıma dönülür.

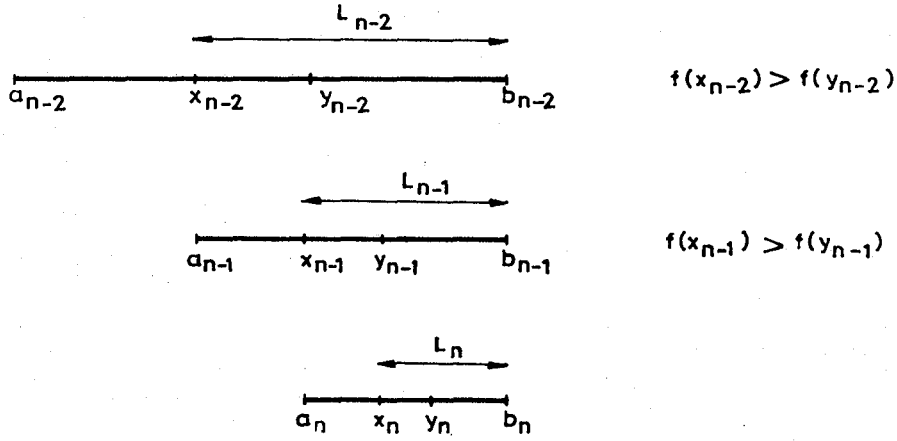
4. Adım : Durulur, x° minimum noktası (a-b) aralığındadır.



Şekil 3.4 : Simetrik İki Nokta yöntemi.

3.9.3. Fibonacci yöntemi

Fibonacci yönteminde de belirsizlik aralığı içinde simetrik iki nokta alınır ancak, aralığın daraltılma miktarı her iterasyonda değişiktir. İlk iterasyonda iki nokta için fonksiyon değeri hesaplanmakla birlikte, sonraki iterasyonlarda yanlızca bir noktada fonksiyon değeri hesaplanır (Şekil 3,5).



Şekil 3.5 : Fibonacci yönteminde aralığı daraltma.

Fibonacci yönteminde işleme başlarken iki simetrik nokta,

$$x_k = a_k + \frac{F_{n-k-1}}{F_{n-k+1}} \cdot (b_k - a_k) \qquad (3\text{-}19)$$

$$y_k = a_k + \frac{F_{n-k}}{F_{n-k+1}} \cdot (b_k - a_k) \qquad (3\text{-}20)$$

ifadelerinden hesaplanır ve yeni belirsizlik aralığı,

$f(x_k) > f(y_k)$ ise $(x_k - b_k)$

$f(x_k) \leq f(y_k)$ ise $(a_k - y_k)$

olarak alınır. F_i katsayıları,

$F_o = F_1 = 1$

olmak koşuluyla,

$$F_{j+1} = F_j + F_{j-1} \quad , \quad (j=1,2,\ldots.) \qquad (3\text{-}21)$$

ifadesinden hesaplanır ve sırasıyla 1,1,2,3,5,7,13,21,.... değerlerini alır (Wilde, 1964).

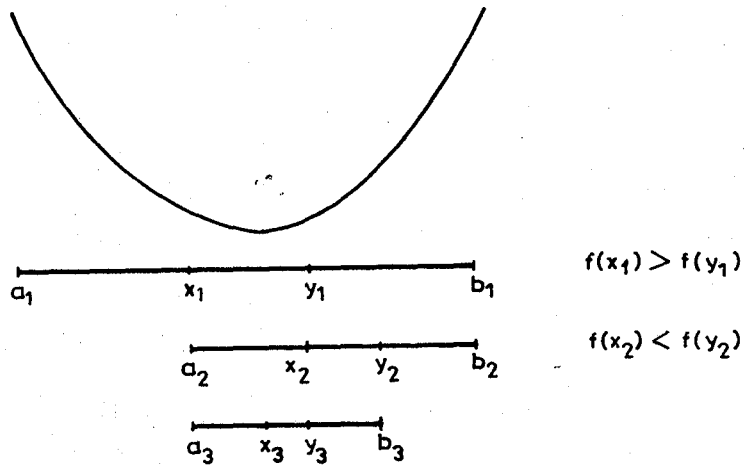
Fibonacci yöntemi aranan noktaya hızlı bir yaklaşım sağlamakla birlikte, yapılacak iterasyon sayısının önceden bilinmesini gerektirir (Avriel, 1976).

3.9.4. Altın Oran yöntemi

Kenarları x ve y olan bir dikdörtgenden y^2 alanlı bir kare parçası çıkartıldığında, arta kalan dikdörtgenin uzun ve kısa kenarlarının oranının, orijinal dikdörtgendeki orana eşit olabilmesi için x ve y boyutları arasında bir ilişki olmalıdır (Gottfried and Weisman,1973).

$$k' = \frac{x}{y} = 1,618034.... \qquad (3\text{-}22)$$

x ve y kenarları arasındaki bu oran eski çağlardan beri bilinir ve "Altın Oran" olarak isimlendirilir (Luenberger, 1973).



Şekil 3.6 : Altın Oran yönteminde aralığın daraltılması.

Altın Oran mantığından belirsizlik aralığının doğrusal aramada optimum bir şekilde daraltılmasında da yararlanılmaktadır (Şekil 3,6).

İşleme başlarken aralığın içinde iki simetrik nokta almak gerekmekle birlikte, sonraki iterasyonlarda tek bir nokta için hesaplama yapmak yeterli olmaktadır.

Yöntemin algoritması Wagner (1969) tarafından aşağıdaki gibi verilmektedir:

1. Adım : (a-b) sınırları ve istenen hassasiyet değeri (ε) verilir,

$$H = b-a \tag{3-23}$$

$$T = 0,5.(\sqrt{5} - 1) \tag{3-24}$$

$$x = a+T^2.H \tag{3-25}$$

$$y = a+T.H \tag{3-26}$$

olarak eşitlenir.

2. Adım : $f(x) \geq f(y)$ ise 3. adıma atlanır,

$f(x) < f(y)$ ise 4. adıma gidilir.

3. Adım : $b = y$, $H = b-a$

$y = x$, $x = a+T^2.H$

alınıp 5. adıma atlanır.

4. Adım : $a = x$, $H = b-a$

$x = y$, $y = a+T.H$

alınıp 5. adıma gidilir.

5. Adım : $H>\varepsilon$ ise 2. adıma dönülür,

$H\leq\varepsilon$ ise durulur, optimum nokta (a-b) aralığındadır.

3.9.5. Kareli İnterpolasyon yöntemi

İnterpolasyon yöntemlerinde belirli noktalarda aldığı değerler kullanılarak fonksiyon, basit bir polinoma yaklaştırılır. Kareli İnterpolasyon yönteminde ikinci dereceden polinoma yaklaşım yapılarak, bu polinomun minimum noktası fonksiyonun minimum noktası olarak nitelenir.

Minimumu aranan f(x) fonksiyonunun α, β, ν noktalarındaki değerleri f_α, f_β, f_υ ise fonksiyon,

$$H(x) = A.x^2 + B.x + C \qquad (3\text{-}27)$$

kareli fonksiyonuna yaklaştırılır.

$$\begin{aligned} f_\alpha &= A.\alpha^2 + B.\alpha + C \\ f_\beta &= A.\beta^2 + B.\beta + C \\ f_\upsilon &= A.\upsilon^2 + B.\nu + C \end{aligned} \qquad (3\text{-}28)$$

eşitliklerinden A, B, C katsayıları,

$$\begin{aligned} A &= [(\nu-\beta).f_\alpha + (\alpha-\upsilon).f_\beta + (\beta-\alpha).f_\upsilon]/\Delta \\ B &= [(\beta^2-\upsilon^2).f_\alpha + (\upsilon^2-\alpha^2).f_\beta + (\alpha^2-\beta^2).f_\upsilon]/\Delta \\ C &= [\beta.\upsilon.(\upsilon-\beta).f_\alpha + \upsilon.\alpha.(\alpha-\upsilon).f_\beta + \alpha.\beta.(\beta-\alpha).f_\upsilon]/\Delta \end{aligned} \qquad (3\text{-}29)$$

olarak belirlenir. Burada Δ katsayısı,

$$\Delta = (\alpha-\beta).(\beta-\upsilon).(\upsilon-\alpha) \qquad (3\text{-}30)$$

eşitliğini ifade etmektedir (Walsh, 1979).

(3-27) kareli fonksiyonunun minimumu,

$$\frac{dH}{dx} = 2Ax + B = 0$$

$$x = -\frac{B}{2A} \qquad (3\text{-}31)$$

noktasında olacaktır. (3-29) katsayıları kullanılarak f(x) fonksiyonunun minimum noktası yaklaşık bir değerle,

$$x^{\circ} = \frac{1}{2}\left[\frac{(\beta^2-\upsilon^2).f_{\alpha} + (\upsilon^2-\alpha^2).f_{\beta} + (\alpha^2-\beta^2).f_{\upsilon}}{(\beta-\upsilon).f_{\alpha} + (\upsilon-\alpha).f_{\beta} + (\alpha-\beta).f_{\upsilon}}\right] \qquad (3\text{-}32)$$

olarak belirlenir.

Ulaşılan bu nokta fonksiyonun gerçek minimum noktası olmayıp, belirli bir uzaklığa düşmektedir. Gerçek minimum noktaya istenen hassasiyette yaklaşmak iteratif işlemleri gerektirir. Her iterasyonda yeni interpolasyonlar yapılarak gerçek minimum noktaya yeterli hassasiyette yaklaşılır.

Yöntemin algoritmasını Bunday (1984) aşağıdaki gibi vermektedir:

1. Adım : Başlangıç noktası D, adım uzunluğu H, istenen hassasiyet değeri (ε) belirlenir.
   Fonksiyonun f(D) ve f(D+H) değerleri hesaplanır.

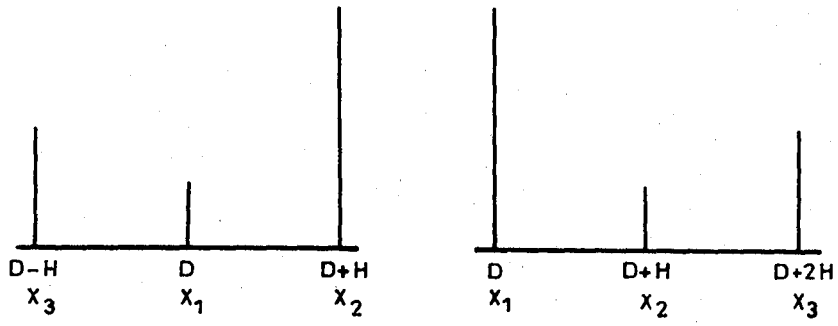
2. Adım : $f(D) < f(D+H)$ ise üçüncü nokta olarak (D-H) alınıp f(D-H) değeri hesaplanır.
   $f(D) \geq f(D+H)$ ise üçüncü nokta olarak (D+2H) alınıp f(D+2H) değeri hesaplanır.

3. Adım : (3-32) eşitliğinden x° noktası belirlenir, $f(x^{\circ})$ değeri hesaplanır.

4. Adım : Fonksiyonun en küçük ve bir sonraki en küçük değerlerini aldığı noktalar arasındaki fark istenen hassasiyetten küçük ise 6. adıma atlanır.

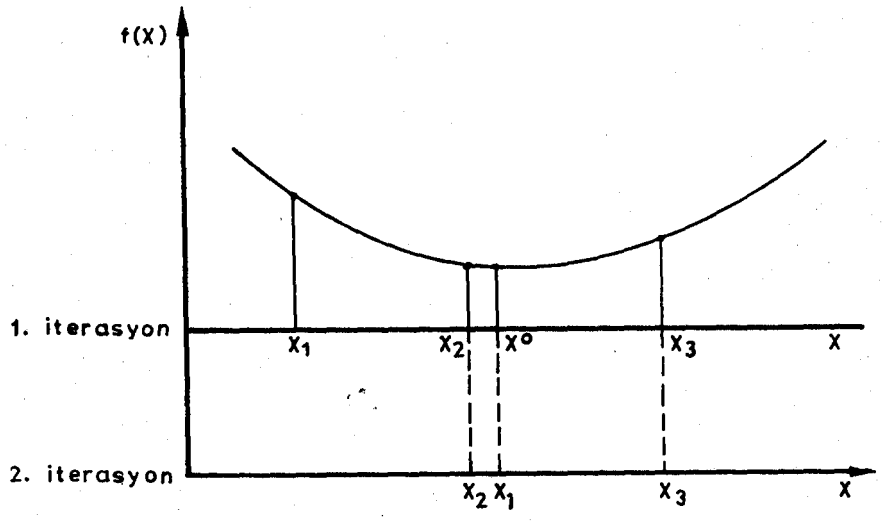
5. Adım : Fonksiyonun en küçük değeri arada kalacak şekilde x noktalarından birisini atarak, arta kalan üç tane x değeri ile 3. adıma gidilir.

6. Adım : Durulur, minimum nokta fonksiyonun en küçük değerini aldığı nokta olarak alınır.



Şekil 3.7 : Kareli İnterpolasyonda noktaların belirlenmesi.

Kareli İnterpolasyon yönteminde fonksiyon değerlerinin hesaplandığı noktaların belirlenmesi Şekil 3,7'de, yöntemin 1. ve 2. iterasyonlardaki işleyişi ise Şekil 3,8'de grafiksel olarak verilmektedir.



Şekil 3.8 : Kareli İnterpolasyon yönteminin işleyişi.

İterasyon işlemlerini bitirmede ölçüt olarak alınan hassasiyet derecesi çok küçük olduğunda ve fonksiyon eğrisinin yayvan bir şekil göstermesi durumunda α, β, υ ve f_α, f_β, f_υ değerleri birbirine çok yakın olacağından x° noktasının hesaplanmasında güçlükler çıkabilir.

Bu güçlüğün giderilebilmesi için, ikinci ve daha sonraki iterasyonlarda (3-32) eşitliği yerine,

$$x^{\circ} = \frac{\alpha+\beta}{2} + \frac{0,5.(f_{\alpha}-f_{\beta})(\beta-\upsilon)(\upsilon-\alpha)}{(\beta-\upsilon)f_{\alpha}+(\upsilon-\alpha)f_{\beta}+(\alpha-\beta)f_{\upsilon}} \qquad (3-33)$$

ifadesinin kullanılması önerilmektedir (Bunday, 1984).

Kareli İnterpolasyon yönteminde tek bir iterasyonda yapılan işlemlerin yoğun olmasına karşın, aranan noktaya bir kaç iterasyonda istenen hassasiyet derecesinde yaklaşılabilmektedir. Fibonacci yöntemindeki gibi yapılacak iterasyon sayısının önceden belirlenmesini ve Kübik Yaklaşım yöntemindeki gibi türev alma işlemlerini gerektirmemesi yönteme üstünlük sağlamaktadır.

4. DOĞRUSAL OLMAYAN MODELLERİN ÇÖZÜM TEKNİKLERİ

4.1. Sınıflama

Doğrusal olmayan modellerin genel matematiksel formuna uyan her problemi çözebilecek genel bir teknik yoktur. Çözülecek problemin amaç fonksiyonu ve kısıtlarının yapılarına göre en uygun çözüm tekniği kullanılır. Doğrusal olmayan programlama çözüm teknikleri aşağıdaki gibi sınıflanabilir (Bradley, et al., 1977):

1. Kısıtsız optimizasyon;

   Amaç fonksiyonu : $f(x_1,x_2,\ldots,x_n)$

2. Doğrusal programlama;

   Amaç fonksiyonu : $f(X) = \sum_{j=1}^{n} C_j.x_j$

   Kısıtlar : $g_i(X)=\sum_{j=1}^{n} a_{ij}.x_j \quad , \quad (i=1,2,\ldots,m)$

   $g_{m+i}(X) = -x_i \quad , \quad (i=1,2,\ldots,n)$

3. Kareli programlama;

   Amaç fonksiyonu : $f(X)=\sum_{j=1}^{n} C_j.x_j + \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}.x_i,x_j$

   Kısıtlar : İkinci maddedeki gibi.

4. Doğrusal kısıtlı problem;

   Amaç fonksiyonu : $f(X)=f(x_1,x_2,\ldots,x_n)$

   Kısıtlar : $g_i(X)=\sum_{j=1}^{n} a_{ij}.x_j \quad , \quad (i=1,2,\ldots,m)$

5. Ayrılabilir programlama ;

   Amaç fonksiyonu : $f(X)=\sum_{j=1}^{n} f_j(x_j)$

Kısıtlar : $g_i(X)=\sum_{j=1}^{n} g_{ij}(x_j)$, $(i=1,2,...,m)$

6. Konveks programlama;

Amaç fonksiyonu : f(X) Enküçükleme için konveks,

Enbüyükleme için konkav fonksiyon.

Kısıtlar : Tümü konveks fonksiyonlar.

4.2. Kısıtsız Modelin Optimizasyonu

Çok değişkenli kısıtsız modelin genel yapısı,

$$U_{\substack{max\\min}} = f(x_1,x_2,...,x_n) \qquad (4\text{-}1)$$

olarak verilir. Bu modelin çözümü için amaç fonksiyonunun ekstremumları araştırılır. Bu araştırmada aşağıdaki algoritma izlenebilir (Kara, 1986):

1. Adım : Amaç fonksiyonunun birinci kısmi türevleri alınarak fonksiyonun gradyanı belirlenir,

$\nabla f(X) = 0$

denklem sisteminin kökleri araştırılır.

a) Bu sistemin kökleri yoksa modelin çözümü yoktur.

b) Sistemin kökleri analitik olarak bulunamıyorsa sayısal çözüm tekniklerine başvurulur.

c) Sistemin kökleri bulunabiliyorsa 2. adıma geçilir.

2. Adım : Sistemi çözen her X^i değeri için Hessien matrisi oluşturularak asal minörlerine bakılır.

a) H pozitif belirli ise X^i noktasında yerel minimum vardır.

b) H negatif belirli ise X^i noktasında yerel maksimum vardır.

c) a ve b durumları oluşmuyorsa bu noktada bir ekstremum yoktur, 3. adıma geçilir.

3. Adım : Amaç fonksiyonunun konveks veya konkav olup olmadığı araştırılır.

a) f(X) konveks ise yerel minimum olan noktada fonksiyon bütünsel enküçük değerini alır.

b) f(X) konkav ise yerel maksimum olan noktada fonksiyon bütünsel enbüyük değerini alır.

c) f(X) konveks veya konkav değilse bütünsel eniyi sözkonusu değildir.

Yeraltı havalandırma şebekelerinin 5. Bölümde verildiği gibi modellenmesi ile doğrusal kısıtlı çok değişkenli bir model oluşturulmuştur. Daha sonra amaç fonksiyonundaki bağımlı değişkenleri bağımsız değişkenler cinsinden ifade ederek, akımın korunumu ilkesini gerçekleyen kısıt fonksiyonları amaç fonksiyonu içine dahil edilmiş ve kısıtsız, çok değişkenli bir model oluşturulmuştur. Böylece literatürde daha güçlü olarak nitelenen kısıtsız optimizasyon tekniklerinin uygulanması mümkün olmuştur.

Havalandırma şebekelerinin optimizasyonunda değişken sayısının fazla olmasından dolayı algoritmanın 1.b durumu ortaya çıkmakta, analitik çözüm olanaksız olmaktadır. Bu durumda sayısal çözüm tekniklerine başvurmak gerekmektedir. Bu nedenle, çok değişkenli kısıtsız modellerin optimizasyonunda etkin olarak kullanılan sayısal çözüm teknikleri ayrıntılı olarak ele alınmıştır.

4.2.1. Sayısal çözüm teknikleri

Çok değişkenli fonksiyonları eniyilemekte kullanılan sayısal çözüm teknikleri, bir başlangıç mümkün çözümünden optimum çözüme doğru ilerleyen tekrarlı işlemleri gerektirir. Bu teknikler amaç fonksiyonunun değişkenlere göre kısmi türevlerini kullanıp kullanmadıklarına

bağlı olarak iki ana gurupta sınıflanabilir (Bazaraa and Shetty, 1979):

1. Türev kullanmayan çok değişkenli arama teknikleri;
   a) Dairesel Koordinat yöntemi,
   b) Hooke-Jeeves yöntemi,
   c) Rosenbrock yöntemi.
2. Amaç fonksiyonunun kısmi türevlerini kullanan çok değişkenli arama teknikleri (Gradyan yöntemleri);
   a) En Hızlı İniş yöntemi,
   b) Newton yöntemi,
   c) Fletcher-Reeves yöntemi,
   d) Davidon-Fletcher-Powell yöntemi,
   e) Zangwill yöntemi.

Amaç fonksiyonunun türevini kullanmayan teknikler "Arama Yöntemleri" (Search Methods) olarak isimlendirilir. Bu tekniklerde bir başlangıç X seti için uygun bir d ilerleme yönü seçilir ve bu yönde amaç fonksiyonu eniyilenir. Bu işlemlerde amaç fonksiyonunun türevleri yerine fonksiyonun kendi değerleri kullanılır (Gottfried and Weisman, 1973).

Arama yöntemlerinde amaç fonksiyonunun düzenli, sürekli ve türevi alınabilir olması gerekmemekte, bu özelliklerinden dolayı türev kullanan yöntemlere göre daha kullanışlı olmaktadırlar. Ancak fonksiyonun en hızlı azalma yönü gözönüne alınmadığından optimum çözüme yakınsama hızları yavaş olmakta, toplam çözüm süresi uzamaktadır (Himmelblau, 1972).

Amaç fonksiyonunun topolojisini değerlendiren ve ilerleme yönü olarak en hızlı değişme yönünü kullanan Gradyan teknikleri, kısıtsız modellerin optimizasyonunda en hızlı işleyen iterasyon teknikleri olarak kabul edilmektedir (Schwefel, 1981 ; Zangwill, 1971).

Havalandırma şebekelerinin modellenmesi ile oluşturulan kısıtsız modelin amaç fonksiyonu sürekli ve türevi alınabilir bir fonksiyondur. Ayrıca ayrılabilir bir fonksiyon olmasından ve kısmi türevlerinin sıfıra eşitlenmesi ile elde edilen denklem sistemlerinin gözlerdeki basınç düşüşlerinin korunumu ilkesini gerçeklemesinden dolayı, amaç fonksiyonunun gradyanı analitik yöntemlere başvurulmaksızın bilgisayarda göz dizilerinin taratılması ile hızlı ve basit bir şekilde hesaplanabilmektedir.

Havalandırma şebekelerinin analizi için uygulanan tekniklerin Hardy Cross yöntemi ile karşılaştırılmasında, çözüm süresi temel ölçüt olarak ele alındığından, en hızlı çözümü veren Gradyan teknikleri havalandırma şebekelerine uygulanmış, arama yöntemleri düşük çözüm hızlarından dolayı analize sokulmamıştır.

4.2.2. Gradyan yöntemleri

Çok değişkenli kısıtsız amaç fonksiyonlarının optimizasyonunda yaygın olarak kullanılan güçlü iterasyon teknikleri "Gradyan Yöntemleri" olarak bilinir. Bu yöntemlerde amaç fonksiyonunun bulunulan noktadaki gradyanı kullanılarak, amaç fonksiyonunu enküçükleyen (veya enbüyükleyen) karar değişkenleri değerlerinin hesaplanmasi ile problem çözülür.

Bir f(X) fonksiyonu sürekli ve türevi alınabilir bir fonksiyon ise, bu fonksiyonun her X noktasında bir $\nabla f(X)$ gradyanı bulunur. Bir X=X' noktasındaki gradyan, elemanları fonksiyonun kısmi türevlerinin X=X''deki değerlerinden oluşan bir vektördür ve o noktadaki eğimi verir (Hillier and Lieberman, 1974).

$$\nabla f(X') = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots\ldots, \frac{\partial f}{\partial x_n}\right) \qquad (4\text{-}2)$$

Grafiksel gösterimi Şekil 4,1'deki gibi olan iki değişkenli bir amaç fonksiyonunun enküçüklenmesi istendiğinde, bir A başlangıç noktasından enküçük noktaya doğru ilerlenmesi gerekir. A noktasından enküçük noktaya ulaşan en kısa yol s yoludur. A başlangıç noktasından başlandığında fonksiyon değerinde en büyük artışı sağlayacak ilerleme yönü o noktada eğimin en fazla olacağı yön, başka bir deyişle gradyanın yönü olacaktır (Şekil 4,2).

Gradyan vektörünün yönü, başlangıç noktasından $\nabla f(X')$ noktasına yönlendirilen doğru parçasının yönü olarak yorumlanır. Bu nedenle X' deki değişimlerin yönü $\nabla f(X)$ ile aynı ise f(X) deki artma oranı en büyük olacaktır. Amaç f(X) i enküçüklemek olduğundan, gradyan vektörünün ters yönünde ilerlemek uygun olacaktır (Abadie, 1967).

4.2.2.1. En Hızlı İniş yöntemi

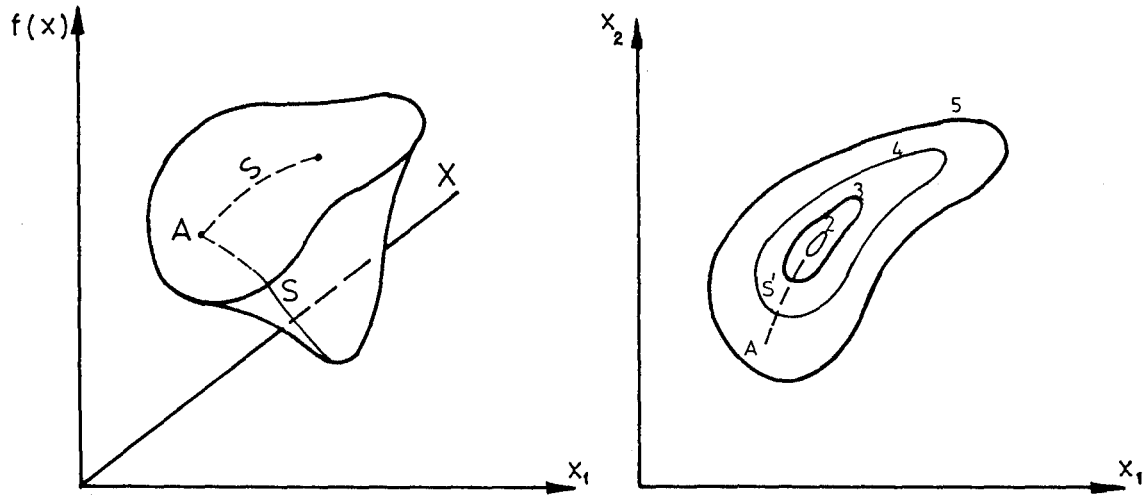
Amaç fonksiyonunun gradyanını kullanan sayısal çözüm tekniklerinin temeli ve en basiti "En Hızlı İniş" (Steepest Descent) yöntemidir. Yöntemin temeli, verilen bir başlangıç noktasından başlayarak fonksiyonun en hızlı azalma yönünde ardışık noktalar bulmaktır (Kowalik and Osborne, 1968).

İşleme herhangi bir X° başlangıç noktasından başlanarak, o noktada fonksiyonun $\nabla f(X^\circ)$ gradyanı belirlenir. X° noktasından gradyanın tersi yönünde hareket edildiğinde, fonksiyon değerindeki azalma miktarı en büyük olacaktır. Bu nedenle ardışık X^1 noktası,

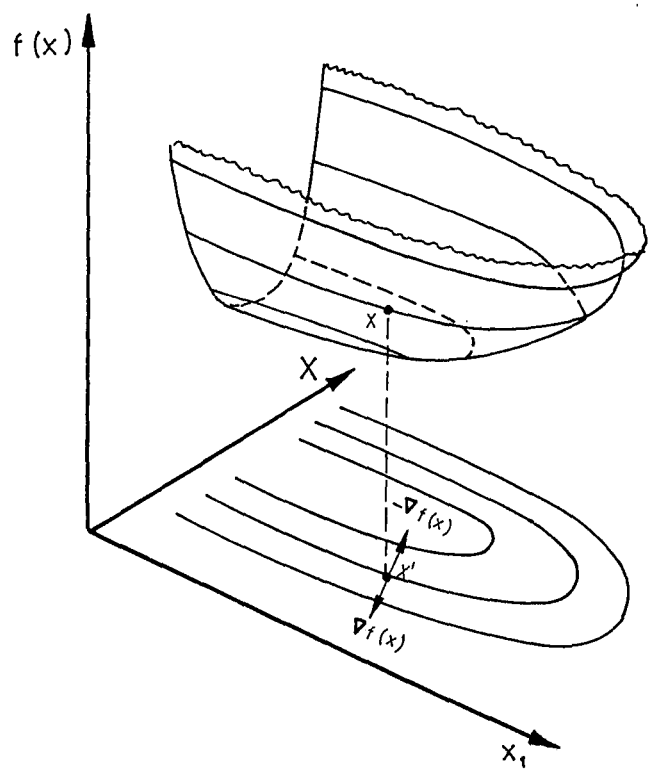
$$X^1 = X^\circ - r.\nabla f(X^\circ) \qquad (4\text{-}3)$$

ifadesine göre belirlenir.

Negatif gradyan vektörü amaç fonksiyonunu enküçüklemek için ilerlenecek yönü belirlemekle birlikte, bu yönde ilerlendiğinde fonk-

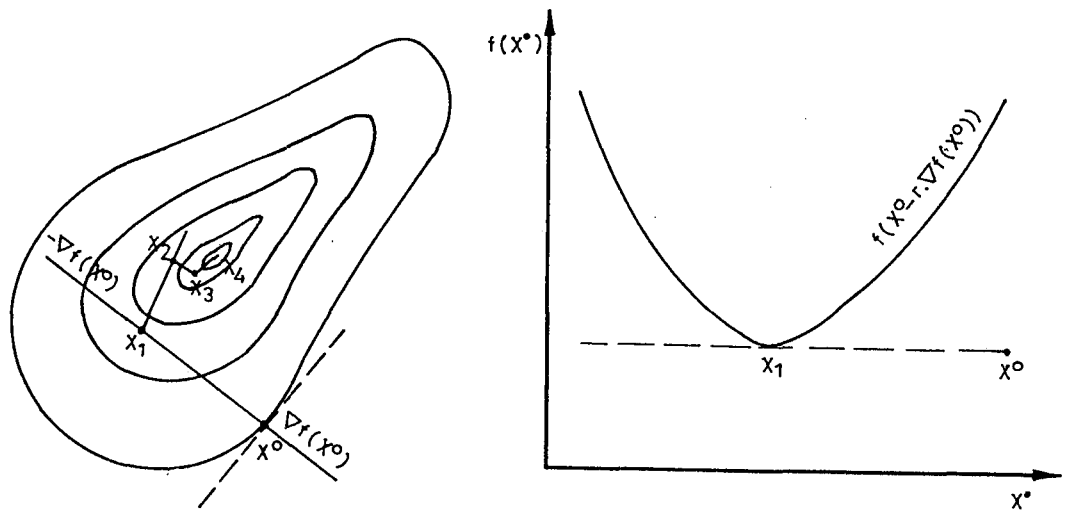


Şekil 4.1 : İki değişkenli bir fonksiyonun enküçüklenmesinde izlenen yol (Gottfried and Weisman, 1973).



Şekil 4.2 : Bir noktadaki gradyan ve en hızlı azalma yönü (Himmelblau, 1972).

siyon değerindeki azalma belli bir noktada duracak ve daha sonra bu değer artmaya başlayacaktır (Şekil 4,3). Bu optimum noktayı belirleyen r paremetresi "Optimal Adım Boyutu" olarak tanımlanır ve (0-1) aralığında pozitif bir sayıdır (Aoki, 1971).



Şekil 4.3 : En Hızlı İniş yönteminde ilerleme.

Şekil 4,3'de geometrik yorumu verildiği gibi, başlangıç noktasından gradyanın tersi yönündeki ilerlemeye fonksiyondaki azalmalar durduğunda son verilerek X^1 noktası belirlenir. X^1 noktasındaki gradyanın tersi yönünde yeniden ilerlenerek X^2 noktasına ulaşılır. İşleme bu şekilde hareket ederek ardışık noktaların bulunması ile devam edilir. Her ardışık nokta,

$$X^{k+1} = X^k - r^k . \nabla f(X^k) \qquad (4\text{-}4)$$

ifadesinden belirlenir. İterasyon işlemlerine,

$$\nabla f(X^k) \approx 0 \qquad (4\text{-}5)$$

olduğunda, başka bir deyişle gradyan vektörü değerlerinin istenen bir hassasiyet derecesi kadar sıfıra yaklaşması durumunda son verilir. Bu koşul amaç fonksiyonunun enküçük olması için gerek şartın belli bir yaklaşıklıkla sağlanmış olduğunu ifade edecek, ulaşılan X^k noktası optimum çözüm setini verecektir.

En Hızlı İniş yönteminin algoritması aşağıda verildiği gibi kısaca özetlenebilir:

1. Adım : X° başlangıç noktası ve istenen hassasiyet değeri (ε) seçilir.

2. Adım : $\nabla f(X^\circ)$ gradyan vektörü hesaplanır.

3. Adım : Ardışık X noktası,

$$X^{k+1} = X^k - r^k.\nabla f(X^k)$$

ifadesince belirlenir.

4. Adım : k=k+1 yapılıp $\nabla f(X^k)$ hesaplanır.

5. Adım : $\nabla f(X^k)<\varepsilon$ ise işlem bitirilir,

Tersi durumda 3. adıma dönülür.

Algoritmanın en güç ve zaman alıcı aşaması r^k optimal adım boyutunun belirlenmesidir. Optimal adım boyutunun her iterasyondaki değeri, gradyanın ters yönündeki doğru parçası üzerinde ilerlendiğinde ardışık X^{k+1} noktası, fonksiyon değerinde en büyük azalmayı sağlayacak şekilde saptanır. Bu değer enküçükleme için,

$$\min f(X^k - r^k.\nabla f(X^k)) \qquad (4\text{-}6)$$

olarak belirlenir (Ortega and Rheinboldt, 1970). Başka bir deyişle, bir h(r) fonksiyonu,

$$h(r) = f(X^k - r^k.\nabla f(X^k)) \qquad (4\text{-}7)$$

olarak tanımlanırsa r^k, h(r) fonksiyonunu enküçükleyen r değeri olur. h(r) fonksiyonu r parametresine bağlı tek değişkenli bir fonksiyon olduğundan, h(r)'yi enküçükleyen r değerini bulma işlemi tek boyutlu arama problemi olarak şekillenir (Taha, 1976).

Optimal adım boyutu,

$$\frac{d}{dr} f(X^k - r.\nabla f(X^k)) = 0 \qquad (4\text{-}8)$$

eşitliğinin en küçük pozitif kökü olarak alınabilir (Luenberger, 1969). Ancak her problem için türev alma yoluyla optimal adım boyutunun belirlenmesi mümkün olmayabilir. Bu durumda (4-7) ifadesini enküçükleyen r değerini hesaplamada tek boyutlu arama yöntemlerinin kullanılması gerekir (Wagner, 1969).

En Hızlı İniş yöntemi basitliğine karşın, literatürde belirtilen bazı olumsuz özelliklere sahiptir;

a) Her iterasyondaki ilerleme yönleri diğerlerinden bağımsız olarak hesaplanmakta ve önceki ilerleme yönlerinin daha sonraki işlemlere bir katkısı olmamaktadır.

b) Optimum noktaya yakınsama hızı büyük oranda fonksiyon eğrisinin tipine bağlıdır. Fonksiyonun Hessien matrisinin maksimum ve minimum değerleri arasındaki oran büyükse, optimum nokta etrafında kısa zigzaglar çizilmekte, gerekli iterasyon sayısı fazla olmaktadır (Fiacco and McCormick, 1968 ; Whittle, 1971 ; Schwefel, 1981).

4.2.2.2. Fletcher-Reeves Yöntemi

En Hızlı İniş yönteminde araştırma vektörlerinin birbirinden bağımsız olmasının yarattığı olumsuzluğu gidermek için, Hestenes ve Steifel (1952) tarafından bileşke yönlerin kullanılmasıyla "Bileşke

Gradyan" (Conjugate Gradient) yöntemi önerilmiş, Fletcher ve Reeves (1964) tarafından yöntem geliştirilmiştir (McCormick, 1983).

Bu yöntemin temeli de, amaç fonksiyonunun kısmi türevleri tarafından belirlenen yönde ardışık olarak tek boyutlu araştırma yapılmasına dayanır. Ancak araştırma vektörü, En Hızlı İniş yönteminde olduğu gibi negatif gradyan vektörüne eşit değildir. Her iterasyondaki araştırma vektörü, bulunulan noktadaki gradyan vektörü ile bir önceki noktadaki araştırma vektörünün bir fonksiyonu olacak şekilde belirlenir. Bu şekilde seçilen ilerleme yönü "Bileşke Yön" (Conjugate Gradient) olarak adlandırılır (Kowalik and Osborne, 1968).

(nXn) boyutlu pozitif belirli bir A matrisi için,

$$P^{i\prime}.A.P^{i+1} = 0 \qquad (4\text{-}9)$$

koşulu sağlanıyorsa P^i ve P^{i+1} yönleri bileşke yönler olarak nitelenir. (Pierre, 1969). İlk adımdaki arama yönü,

$$P^o = -\nabla f(X^o) \qquad (4\text{-}10)$$

olarak alındıktan sonra bir sonraki adımda bileşke yön,

$$P^{i+1} = -\nabla f(X^{i+1}) + \beta_i.P^i \qquad (4\text{-}11)$$

olarak belirlenir. Burada β_i, hesaplanması gereken pozitif bir parametredir ve (4-9) ile (4-11) eşitliklerinden,

$$-P^{i\prime}.A.\nabla f(X^{i+1}) + \beta_i.P^{i\prime}.A.P^i = 0$$

$$\beta_i = \frac{P^{i\prime}.A.\nabla f(X^{i+1})}{P^{i\prime}.A.P^i} \qquad (4\text{-}12)$$

olarak elde edilir. (4-12) eşitliği β_i parametresinin hesaplanmasında

kullanılabilir. Ancak bu eşitlik A matrisinin bilinmesini, bilgisayar belleğinde (nXn) boyutlu yerin kullanılmasını gerektirir ve kareli biçimdeki fonksiyonlar için kullanılabilir. Kareli biçimde olmayan fonksiyonlar için de kullanılabilen ve A matrisini gerektirmeyen daha basit bir ifade,

$$\beta_i = \frac{\nabla f(X^{i+1})'.\nabla f(X^{i+1})}{\nabla f(X^i)'.\nabla f(X^i)} \qquad (4\text{-}13)$$

şeklinde verilmektedir (Gottfried and Weisman, 1973). Bu ifadeye dayanarak hesaplanan β_i değerinin (4-11) eşitliğine uygulanması ile P^i vektörünün bileşke yönü olan P^{i+1} belirlenmekte ve bu yönde ilerlenerek ardışık nokta,

$$X^{i+1} = X^i + r^i.P^{i+1} \qquad (4\text{-}14)$$

eşitliğinden hesaplanmaktadır (Fletcher and Reeves, 1964).

Optimal adım boyutunun değeri En Hızlı İniş yöntemindekine benzer şekilde,

$$\min f(X^i + r^i.\ P^i) \qquad (4\text{-}15)$$

olacak biçimde P^{i+1} yönünde tek boyutlu araştırma yapılmasıyla saptanmaktadır (Pierre, 1969).

İterasyon işlemlerine ardışık noktaların bu şekilde bulunması ile devam edilir ve (4-5) şartı gerçeklendiğinde, yani ulaşılan noktadaki gradyan belli bir yaklaşıklıkla sıfır olduğunda son verilir. Bu koşulu gerçekleyen X^i noktası amaç fonksiyonunu enküçükleyen çözüm setini verecektir.

Fletcher-Reeves yöntemi En Hızlı İniş yöntemine oranla çok daha hızlı bir yakınsama sağlamakla birlikte, yuvarlatma hatalarına karşı oldukça duyarlıdır (Gottfried and Weisman, 1973). Yuvarlatma hatala-

rını en aza indirebilmek için Fletcher ve Reeves (1964), her (n+1) iterasyondan sonra hesaplamaya $P^i=-\nabla f(X^i)$ yönünde yeniden başlamayı önermişlerdir.

Yöntemin algoritması aşağıdaki gibi özetlenebilir (Kunzi and Krelle, 1966);

1. Adım : X° başlangıç noktası ve istenen hassasiyet miktarı (ε) seçilir, $j=0$ alınır.

2. Adım : $P^\circ = -\nabla f(X^\circ)$ olarak atanır.

3. Adım : $r^i = \min f(X^i + r^i.P^i)$ olacak şekilde,

$X^{i+1} = X^i + r^i.P^i$ hesaplanır,

$|\nabla f(X^{i+1})|<\varepsilon$ ise işlem bitirilir.

4. Adım : $j=j+1$ yapılır,

$j=n$ ise $P^i=-\nabla f(X^i)$ ve $j=0$ yapılıp 3. adıma dönülür.

$j\neq n$ ise $P^{i+1}=-\nabla f(X^{i+1})+\beta_i.P^i$ olarak alınır ve 3. adıma dönülür. Buradaki β_i parametresi (4-13) ifadesinden saptanır.

Algoritmanın n bağımsız değişkenli kareli bir fonksiyonu en çok n iterasyonda enküçüklemeyi garanti ettiği belirtilmektedir (Kunzi and Krelle, 1966 ; McCormick, 1983).

Fletcher ve Reeves (1964), önerdikleri yöntemin Davidon-Fletcher-Powell yöntemine oranla çözüme daha fazla iterasyonda ulaşabildiğini, ancak bu ikinci yöntemin her iterasyonunda yapılan işlemlerin çok daha kompleks olduğunu ve bilgisayar belleğinde daha fazla yer kullandığını belirterek, gerek toplam çözüm süresi, gerekse kullanılan bellek hacmı bakımından, geliştirdikleri yöntemin çok daha üstün olduğunu ileri sürmüşlerdir.

4.2.2.3. Newton yöntemi

Newton yöntemi denklemlerin yaklaşık köklerini bulmak için kullanılan en yaygın yöntemlerden birisidir. Bir x_1 başlangıç noktasından fonksiyona çizilen teğetin x eksenini kestiği noktadan başlayarak ekstrapolasyon yapılır ve bulunan her değer bir sonraki basamak için kullanılır (Şenel, 1983).

Newton yöntemi fonksiyon enküçüklenmesinde de kullanılabilir. Enküçüklenmesi istenen tek değişkenli bir f(x) fonksiyonunda, bir x_k başlangıç noktası için $f(x_k)$, $f'(x_k)$, $f''(x_k)$ değerleri hesaplanır. Aranan gerçek enküçük nokta x, gözönüne alınan x_k noktasının bu noktadan sapması h ise,

$$x = x_k + h \tag{4-16}$$

olur. Bu ifadenin fonksiyondaki değeri Taylor serisine açılırsa,

$$f(x+h) = f(x_k) + f'(x_k)(x-x_k) + \tfrac{1}{2} f''(x_k)(x-x_k) + \ldots\ldots = q(x) \tag{4-17}$$

eşitliği elde edilir. Ardışık x_{k+1} noktasının enküçük olabilmesi için,

$$q'(x) = 0 \tag{4-18}$$

olmalıdır. Taylor açılımının ikinci dereceden ve daha yüksek dereceli terimleri ihmal edilip türev alınırsa,

$$q'(x_{k+1}) = f'(x_k) + f''(x_k)(x_{k+1}-x_k) = 0 \tag{4-19}$$

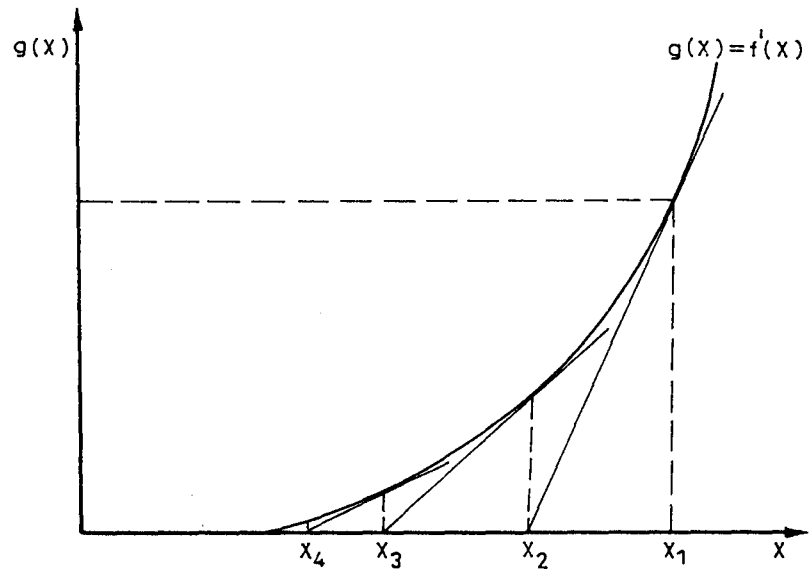
bulunur. Buradan x_{k+1} çekilerek,

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)} \tag{4-20}$$

ifadesi elde edilir.

Bu iterasyon işleminin $q'(x_{k+1}) \approx 0$ olana dek sürdürülmesi ile fonksiyonun enküçük noktasına belli bir yaklaşıklıkla ulaşılmış olur (Luenberger, 1973).

Tek değişkenli bir fonksiyonun enküçük noktasının belirlenmesinde Newton yöntemi ile elde edilen yakınsamanın geometrisi Şekil 4,4'de verilmektedir.



Şekil 4.4 : Newton yönteminde enküçük noktaya yaklaşım (Ortega and Rheinboldt, 1970).

Newton yöntemi çok değişkenli fonksiyonların enküçüklenmesinde de benzer şekilde kullanılabilir. Yöntemin temeli, fonksiyonu her iterasyonda kareli bir fonksiyona yaklaştırmak ve bu kareli fonksiyonun enküçük noktasına doğru ilerlemektir (Walsh, 1975). Başka bir deyişle, gradyan yöntemlerinde amaç fonksiyonuna doğrusal yaklaştırma yapılırken, Newton yönteminde kareli yaklaştırma yapılmakta, ikinci kısmi türevler kullanılmaktadır (Himmelblau, 1972).

n değişkenli bir fonksiyon X^k noktası civarında Taylor serisine açılırsa,

$$q(X) \simeq f(X^k) + \nabla f(X^k)(X-X^k) + \tfrac{1}{2}\,(X-X^k)'.H(X^k).(X-X^k) + \ldots. \qquad (4\text{-}21)$$

biçiminde ifade edilen, X^k noktasında fonksiyonun kareli yaklaşımı elde edilir (Bazaraa and Shetty, 1979). Bu ifadede $H(X^k)$ fonksiyonun X^k noktasındaki Hessien matrisidir. Bu eşitliğin enküçüklenmesi için gerek şart,

$$\nabla q(X) = 0$$

$$\nabla f(X^k) + H(X^k).(X-X^k) = 0 \qquad (4\text{-}22)$$

olmasıdır. X eniyi noktasına yaklaşan X^{k+1} noktası gözönüne alınırsa ardışık nokta,

$$X^{k+1} = X^k - H^{-1}(X^k).\nabla f(X^k) \qquad (4\text{-}23)$$

olarak verilir (Aoki, 1971).

Amaç fonksiyonunun kareli biçimde olması durumunda eniyi noktaya tek bir adımda ulaşılabilmekle birlikte, genel fonksiyonlar için eniyi noktaya yaklaşmak, verilen eşitliğin tekrarlı olarak uygulanmasını gerektirir. Her iterasyonda bulunulan noktadaki gradyan vektörü ve Hessien matrisi hesaplanmalıdır. İterasyon işlemlerine (4-5) koşulu gerçeklenene kadar devam edilir.

Newton yöntemi literatürde kısıtsız konveks fonksiyonların enküçüklenmesinde en güçlü yöntemlerden birisi olarak anılmakla birlikte, aşağıdaki sakıncalarına da işaret edilmektedir;

a) Hessien matrisinin inversi her zaman var olmayabilir,
b) İkinci kısmi türevlerin alınması zor ve zaman alıcıdır,
c) Bilgisayar belleğinde fazla yer işgal edilir.

Özellikle her iterasyonda Hessien matrisinin oluşturulmasının ve bunun inversinin alınmasının güç ve zaman alıcı olması, bellekte

fazla yer kullanması nedeniyle büyük boyutlu problemler için Newton yönteminin kullanımının kısıtlı olduğu rapor edilmektedir (Fiacco and McCormick, 1968 ; Avriel, 1976 ; Aoki, 1971 ; Schwefel, 1981 ; Dennis and Schnabel, 1983 ; Simmons, 1975).

4.2.2.4. Davidon-Fletcher-Powell yöntemi

Kısaca D-F-P yöntemi olarak isimlendirilen Davidon-Fletcher-Powell yöntemi Davidon (1959) tarafından önerilmiş, Fletcher ve Powell (1963) tarafından geliştirilmiştir. Sürekli ve türevi alınabilir kısıtsız amaç fonksiyonlarının eniyilenmesinde kullanılan güçlü bir iterasyon tekniğidir (Murtagh, 1970).

Yöntemin temeli, Newton yönteminde Hessien matrisinin oluşturulması ve inversinin alınmasının sakıncalarını gidermek için H^{-1} yerine, pozitif belirli simetrik bir H matrisi atamak ve bu matrisi her iterasyonda H^{-1} matrisine yaklaştırmaktır (Fox, 1973). Başka bir deyişle yöntem ikinci kısmi türevleri ve invers alma işlemlerini kullanmaksızın, n iterasyonda $H^{-1}(X)$ matrisini oluşturur (Pierre, 1969).

Diğer gradyan yöntemlerinde olduğu gibi araştırmaya bir X^o başlangıç noktasından başlanarak ardışık noktalar,

$$X^{i+1} = X^i + r^i.P^i \quad , \quad (i=0,1,2,\dots) \qquad (4\text{-}24)$$

ifadesine göre hesaplanır. Optimal adım boyutunun belirlenmesinde yine tek boyutlu arama teknikleri kullanılır.

Davidon-Fletcher-Powell yöntemi algoritmasının diğer gradyan yöntemlerinden farkı, P^i araştırma vektörlerinin oluşturulmasındadır. Araştırma vektörlerinin saptanmasında,

$$P^i = -H_i.\nabla f_i \qquad (4\text{-}25)$$

eşitliği kullanılır. Burada H_i pozitif belirli, (nXn) boyutlu simetrik bir matristir. i=0 için H_o başlangıç matrisi, pozitif belirli herhangi bir matris olması koşuluyla rassal olarak atanır. Sonraki adımlarda ise H_i matrisi ardışık olarak hesaplama yoluyla belirlenir (McCormick, 1983).

P^i araştırma vektörlerinin bu şekilde seçilmesi ile,

$$P^{i'}.H.P^j = 0 \qquad (4\text{-}26)$$

olarak verilen bileşke yön koşulu sağlanmış olmaktadır (Pearson, 1969). Ayrıca, H_i matrisinin simetrik ve pozitif belirli bir matris olarak oluşturulması durumunda, aşağıda verilen algoritmanın uygulanması ile türetilecek H_{i+1} matrisi de simetrik ve pozitif belirli bir matris olmaktadır (Fletcher and Powell, 1963).

Böylece H_o başlangıç matrisinin simetrik ve pozitif belirli bir matris biçiminde seçilmesi durumunda, her iterasyonda kullanılan P^i araştırma yönü, bulunulan nokta ile bir önceki noktanın bileşke yönü olacaktır.

H_o başlangıç matrisinin simetrik, pozitif belirli bir matris olmasını sağlamanın en basit yolu, H_o matrisini (nXn) boyutlu birim matris olarak atamaktır.

$$H_o = I \quad , \quad (i=0 \text{ için}) \qquad (4\text{-}27)$$

Bu atama ile birinci iterasyonda En Hızlı İniş yöntemi uygulanmış olmaktadır (Kowalik and Osborne, 1968).

Algoritmanın sonraki adımlarında H_i matrisi,

$$H_{i+1} = H_i + Y_i + Z_i \quad , \quad (i=0,1,2,\ldots.) \qquad (4\text{-}28)$$

eşitliğinden hesaplanır. Buradaki Y ve Z matrisleri (nXn) boyutlu

simetrik matrislerdir ve her iterasyonda aşağıda verilen eşitliklerden hesaplanırlar (Gottfried and Weisman, 1973 ; Zoutendijk, 1970):

$$Y_i = \frac{r^i.p^i.p^{i'}}{p^{i'}.(\nabla f_{i+1}-\nabla f_i)} \qquad (4\text{-}29)$$

$$Z_i = -\frac{H_i.(\nabla f_{i+1}-\nabla f_i).(\nabla f_{i+1}-\nabla f_i)'.H_i'}{(\nabla f_{i+1}-\nabla f_i)'.H_i.(\nabla f_{i+1}-\nabla f_i)} \qquad (4\text{-}30)$$

H_i matrislerinin bu şekilde oluşturularak P^i araştırma vektörlerinin hesaplanması ile bileşke yön üzerinde hareket edilmekte ve bu algoritma n bağımsız değişkenli kareli bir fonksiyonun en fazla n iterasyonda eniyilenmesini garantilemektedir.

İterasyon işlemleri bitirildiğinde H_n matrisi, fonksiyonun optimum noktadaki Hessien matrisinin inversine yaklaşık olarak eşitlenmiş olmaktadır.

$$H_n = H^{-1} \qquad (4\text{-}31)$$

En Hızlı İniş yöntemi basitliğine karşın, yavaş bir yakınsama hızı göstermekte, Newton yöntemi ise yüksek yakınsama hızına karşın yoğun matris işlemlerini gerektirmekte, ayrıca ilk iterasyonda optimum noktanın çok uzaklarına düşebilmektedir. D-F-P yöntemi bu iki klasik yöntem arasında bir denge kurmaktadır (Murtagh, 1970).

Bir başka deyimle, yöntem ilk iterasyonda En Hızlı İniş yöntemini izlemekte, optimum noktanın yakınlarında ise Newton yöntemi gibi çalışarak bu yöntemi yüksek yakınsama hızına uymaktadır (Walsh, 1975).

Fletcher ve Powell (1963) uyguladıkları sayısal örneklere dayanarak, önerdikleri yöntemin diğer gradyan tekniklerine oranla daha üstün olduğunu, büyük boyutlu problemler için de başarıyla kullanılabi-

leceğini ileri sürmüşlerdir. Bu tez Himmelblau (1972), Pearson (1969), Leon (1966) ve Luenberger (1965) tarafından alınan sonuçlarla da desteklenmektedir. Zoutendijk (1970) de yöntemin optimum çözüme daha az iterasyonda ulaşabildiğini, ancak her iterasyonda harcanan sürenin fazla olmasından dolayı toplam çözüm süresinin yüksek olabileceğini belirtmektedir.

5. HAVALANDIRMA ŞEBEKELERİNİN OPTİMİZASYONUNDA MATEMATİKSEL MODELİN OLUŞTURULMASI

5.1. Problemin Tanımı

Doğal dağılımlı havalandırma şebekelerinin analizinde problem, bir veya birkaç basınç kaynağı bulunan N kollu bir şebekede, kollardaki hava dağılımının hesaplanması olarak şekillenir. Hava dağılımını belirleyen parametreler kolların dirençleri ve şebekedeki vantilatörlerin çalışma koşullarıdır. Kol dirençleri basınç düşürücü, vantilatörler ise basınç yaratıcı rol oynarlar.

Böylece optimizasyon problemi bu parametrelere bağlı olarak, kavşaklarda akımın korunumu ilkesini gerçekleme kısıtı altında, şebekedeki toplam basınç düşüşünü veya toplam güç tüketimini enküçükleyen hava dağılımı değerlerinin hesaplanması olarak tanımlanır.

5.2. Amaç Fonksiyonunun Oluşturulması

Bir hava yolunun başlangıç ve bitiş noktaları arasında havanın hareketi sırasında oluşan basınç düşüşü, o koldaki sürtünme kaybı, şok kaybı, doğal hava basıncı ve vantilatör basıncının bir fonksiyonu olarak (2-1) eşitliği ile ifade edilir.

Sürtünme kaybı için Atkinson eşitliği yerine yazılıp, şok kaybı ve doğal hava basıncının etkisi ihmal edilirse,

$$h = R.Q^2 - h_v \qquad (5\text{-}1)$$

ifadesi elde edilir. Sözkonusu kolda bir vantilatör bulunuyorsa, vantilatör tarafından yaratılan h_v basınç farkı, vantilatör karakteristik

eğrisinin bilinen hava miktarı-yük değerlerinden, ifade ettiği ikinci dereceden polinomun katsayılarının belirlenmesi ile,

$$h_v = A + B.Q + C.Q^2 \qquad (5\text{-}2)$$

şeklinde matematiksel olarak gösterilebilir (Wang, 1984). A, B, C katsayıları belirli bir vantilatör için, belirli çalışma şartlarında sabit katsayılardır. Bu durumda vantilatörün yaratacağı basınç farkı, bulunduğu koldan geçen hava miktarının fonksiyonu olmaktadır. Böylece (5-1) eşitliği,

$$h = R.Q^2 - (A + B.Q + C.Q^2) \qquad (5\text{-}3)$$

durumuna gelir.

Vantilatör bulunmayan kollardaki basınç düşüşü ise, şok kaybı ve doğal hava basıncının etkisinin ihmal edilmesi koşuluyla, o koldaki sürtünme kaybına eşit olacaktır.

Verilen eşitlikler şebekenin bir tek kolundaki basınç düşüşünü ifade etmekte, şebekeyi oluşturan tüm kollardaki basınç düşüşlerinin toplamı ise şebekenin toplam basınç düşüşüne eşit olmamaktadır. Bir başka deyişle, şebekenin toplam basınç düşüşü fonksiyonu ayrılabilir bir fonksiyon değildir. Bu nedenle, amaç fonksiyonu olarak kollardaki basınç düşüşlerinin toplamını ifade eden bir fonksiyonun alınması ve enküçüklenmesi olanaklı olmamaktadır.

Bir koldaki basınç düşüşünü tanımlayan (5-3) eşitliğinin Q değişkenine göre integrali alınırsa,

$$U = \int_0^Q (R.Q^2 - (A + B.Q + C.Q^2)) \; . \; dQ \qquad (5\text{-}4)$$

$$U = \frac{1}{3} R.Q^3 - (A.Q + \frac{1}{2} B.Q^2 + \frac{1}{3} C.Q^3) \qquad (5\text{-}5)$$

eşitliği elde edilir. Ele alınan kolda vantilatör yoksa bu ifade,

$$U = \frac{1}{3} R.Q^3 \qquad (5-6)$$

durumuna gelmektedir. Bu eşitlik gözönüne alınan kolun karakteristik eğrisinin altında kalan alanı tanımlamakta ve o kolda tüketilen hava gücünün üçte birine eşit olmaktadır.

Havalandırma şebekelerinde kollardaki basınç düşüşlerinin toplamı şebekenin toplam basınç düşüşüne eşit olmamakla birlikte, kollardaki güç tüketimlerinin toplamı şebekenin toplam güç tüketimini vermektedir. Bu özellikten faydalanılarak, amaç fonksiyonu için kollardaki güç tüketimlerinin toplamını ifade eden bir matematiksel fonksiyon alınabilir. Amaç fonksiyonunun değişkenlere göre kısmi türevleri alındığında oluşan denklem sistemlerinin Kirchoff II yasasını gerçeklemesi bakımından amaç fonksiyonunun elemanları olarak, bir kolda tüketilen hava gücünün üçte birini veren (5-6) eşitliğinin alınması daha uygun olmaktadır.

Bu durumda kol sayısı N olan bir şebeke için amaç fonksiyonu,

$$U = \frac{1}{3} \sum_{i=1}^{N} R_i.Q_i^3 \qquad (5-7)$$

olarak belirlenir. Kol sayısı N, vantilatör sayısı W kadar olan bir şebeke için ise amaç fonksiyonu,

$$U = \frac{1}{3} \sum_{i=1}^{N} R_i.Q_i^3 - \sum_{j=1}^{W} (A_j.Q + \frac{1}{2} B_j.Q^2 + \frac{1}{3} C_j.Q^3) \qquad (5-8)$$

şeklinde olmaktadır. Bu yapıdaki amaç fonksiyonu, Q_i değerlerinin negatif işaretli olabilmesinden dolayı konveks olmayabilir. Bu durum, uygulanacak çözüm tekniklerini sınırlar ve çözüm işlemlerini güçleştirir. Amaç fonksiyonunun konveks bir fonksiyon olmasını sağlamak için, kübik ifadenin mutlak değeri kullanılırsa amaç fonksiyonunun son yapısı,

$$U = \frac{1}{3} \sum_{i=1}^{N} (R_i . |Q_i^3|) - \sum_{j=1}^{W} (A_j . Q + \frac{1}{2} B_j . Q^2 + \frac{1}{3} C_j . Q^3) \qquad (5\text{-}9)$$

biçiminde oluşturulmuş olur. Problem, bu amaç fonksiyonunu enküçükleyecek Q_i değişkenlerinin hesaplanması olarak şekillenir.

5.3. Modelin Kısıtları

Oluşturulan amaç fonksiyonu matematiksel olarak doğrusal olmayan bir ilişkiyi göstermektedir. Bu nedenle, amaç fonksiyonunu enküçükleyen Q_i değerlerinin hesaplanmasında doğrusal olmayan programlama teknikleri kullanılır. Ancak ulaşılan çözümün optimum çözüm olabilmesi için havalandırma şebeke teorisinin öngördüğü, akımın korunumu kısıtlarının gerçeklenmesi de gerekir.

Kısıtların da dahil edilmesiyle doğrusal olmayan programlama probleminin modeli kol sayısı N, kavşak sayısı K kadar olan bir şebeke için;

Amaç fonksiyonu :

$$U \rightarrow \text{Enküçüklenecek,} \qquad (5\text{-}10)$$

Kısıtlar :

$$\sum_{j=1}^{N} e_{ij} . Q_j = 0 \quad , \quad (i=1,2,\ldots,K)$$

olarak şekillendirilmiş olur. Burada e_{ij} şebekenin (KXN) boyutlu durum matrisidir. E durum matrisinin elemanları;

j kolu i kavşağına bağlanmıyorsa ..... $e_{ij} = 0$

j kolu i kavşağından akıyorsa ..... $e_{ij} = 1$

j kolu i kavşağına doğru akıyorsa ..... $e_{ij} = -1$

kodlaması ile oluşturulur (Phillips, et al., 1981).

EK-1'de verilen Örnek Şebeke-1 için durum matrisi;

$$E = \begin{vmatrix} 1 & 1 & 0 & 0 & 0 & 0 & -1 \\ -1 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{vmatrix}$$

(5X7) boyutlu E matrisi, enküçüklenecek amaç fonksiyonu ise;

$$U = \frac{1}{3}\,(R_1.|Q_1^3| + \ldots + R_7.|Q_7^3|) - (A.Q_6 + \frac{1}{2}\,B.Q_6^2 + \frac{1}{3}\,C.Q_6^3)$$

eşitliği ile ifade edilecek, modelin kısıtları ise akımın korunumu ilkesine göre;

$$Q_7 - Q_1 - Q_2 = 0$$
$$Q_4 - Q_1 - Q_3 = 0$$
$$Q_2 - Q_3 - Q_5 = 0$$
$$Q_6 - Q_4 - Q_5 = 0$$
$$Q_7 - Q_6 = 0$$

yapısında oluşturulacaktır.

Modelin amaç fonksiyonu doğrusal olmayan bir fonksiyon olmakta, kısıt fonksiyonları ise doğrusal ilişkileri ifade etmektedir. Bu nedenle havalandırma şebekelerinin analizinde, kısıtları doğrusal ve eşitlik halinde olan modellerin enküçüklenmesinde kullanılan doğrusal olmayan programlama teknikleri, oluşturulan modele uygulanabilir.

5.4. Kısıtsız Model

Uygun teknikler kullanılarak, yapısında kısıtlar içeren modelin optimum çözüm setine ulaşılabileceği gibi, bazı durumlarda kısıt fonksiyonlarını amaç fonksiyonu içine sokarak kısıtsız bir model oluşturmak da olanaklı olabilmektedir. Bu tür modellere uygulanan kısıtsız optimizasyon teknikleri daha basit yapılı olmakta ve optimum çözüme

daha hızlı bir şekilde ulaşılabilmektedir (Fox, 1960 ; Simmons, 1975 ; Phillips, et al., 1976).

Şebeke Analizi teorisine göre kol sayısı N, kavşak sayısı K olan bir şebekede N kadar akım bulunur. Ancak bu akımların tümü bağımsız olmayıp, Kirchoff I yasasına uygun olarak birbirleri ile ilişkilidirler. Bu ilişkinin formüle edilmesi ile (K-1) adet kol akımı, diğer kol akımları cinsinden ifade edilebilir. Başka bir deyişle, şebekedeki (K-1) adet akım bağımlı değişken olmakta,

$$G = N - (K - 1) = N - K + 1 \qquad (5\text{-}11)$$

adet akım ise bağımsız değişken olarak nitelenmektedir (Jensen and Barnes, 1980).

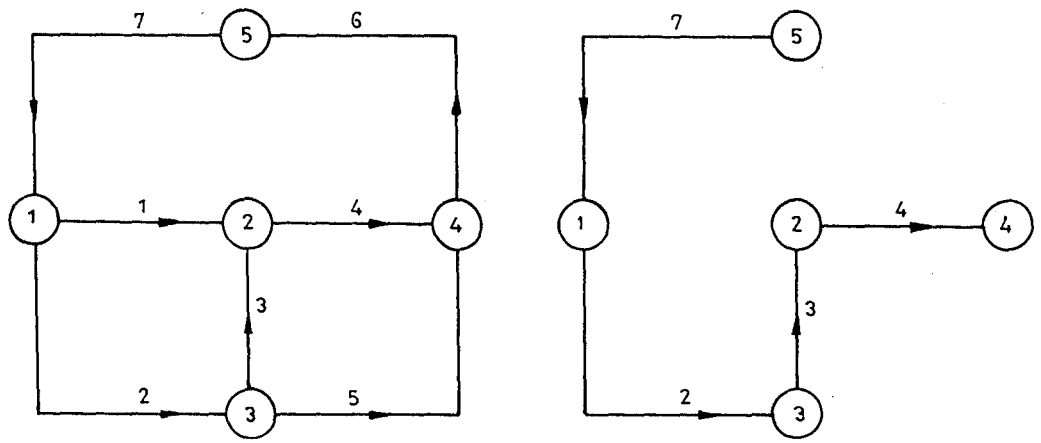
Kısıtlı modelin kısıt fonksiyonları önceki bölümde verildiği gibi, Kirchoff I yasasının matematiksel ifadesi olmaktadır. Şebekeyi oluşturan kollardaki akımları uygun işlemlerle bağımlı ve bağımsız akımlar olarak ayırarak ve bağımlı akımları diğerleri cinsinden ifade ederek modeldeki kısıt sayısı (K-1)'e indirilebilir. Bağımlı değişkenlerin bu ifadeleri amaç fonksiyonunda yerine konularak kısıtlar ortadan kaldırılabilir. Bu düzenleme ile amaç fonksiyonundaki bağımlı değişkenler amaç fonksiyonunun yapısı içinde diğerleri cinsinden ifade edilmiş olacak, akımın korunumu koşulu kısıt fonksiyonları oluşturmaksızın gerçeklenmiş olacaktır.

Şebeke Analizi teorisinde kavşak sayısı K olan bir şebekede (K-1) adet kol, kapalı bir göz yaratmayacak ve tüm kavşaklardan geçecek şekilde birleştirilerek "Ağaç" (Tree) oluşturulur (Ford and Fulkerson, 1960).

Oluşturulan ağacın kolları "Tali kol", arta kalan G adet kol ise "Ana kol" olarak nitelenir. Ana kollardaki akımlar bağımsız, tali kol-

lardaki akımlar ise bağımlı akım olur.

Örnek Şebeke-1 için oluşturulan ağaç, Şekil 5,1'deki gibi olacak, 1-5-6 kolları ana kol, 2-3-4-7 kolları ise tali kol olarak alınacaktır.



Şekil 5.1 : Örnek Şebeke-1 için oluşturulan ağaç,
a) Şebeke b) Ağaç.

Tali kollardaki akımları ana kollardakiler cinsinden ifade edebilmek için göz tekniğinden yararlanılabilir. Bu teknikte bir ana koldan başlanarak, kapalı bir göz oluşturacak şekilde tali kollar dizisi seçilir. Saat akrebinin dönüş veya tersi yönünde bir ilerleme yönü pozitif yön olarak kabul edilir. Göz seçiminde ele alınan kol akımı bu yönde ise kol numarası pozitif, ters yönde ise negatif işaretli olarak alınır. Benzer şekilde,her gözde bir tane ana kol bulunması ve bir ana kolun sadece bir gözde kullanılması koşuluyla G adet göz seçilir.

Verilen ilkelere uygun olarak Örnek Şebeke-1'den seçilen gözler;

GÖZ 1 : 6 7 2 3 4

GÖZ 2 : 1 -3 -2

Göz 3 : 5 -4 -3

düzeninde olacaktır.

Bir şebeke için seçilen göz dizileri (GXN) boyutlu bir D matrisi ile matris formunda ifade edilebilir. D göz matrisinin elemanları;

j kolu i. gözde bulunmuyorsa ..... $d_{ij} = 0$

j kolu i. gözde pozitif yönlü ise ... $d_{ij} = 1$

j kolu i. gözde negatif yönlü ise ... $d_{ij} = -1$

olarak alınır. Örnek Şebeke-1 için göz matrisi;

$$D = \begin{vmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 1 & 0 & 0 \end{vmatrix}$$

formunda olacaktır.

Bir şebekenin göz matrisi,

$$D = \left| I_g \quad D_{it} \right| \tag{5-12}$$

olarak parçalanabilir. Burada;

I_g : Ana kollara ait (GXG) boyutlu birim matris,

D_{it}: Tali kollara ait (GX(K-1)) boyutlu göz alt matrisidir.

Göz alt matrisini kullanarak tali kollardaki akım miktarları ana kollardaki akımlar cinsinden,

$$Q_t = \sum_{i=1}^{G} d_{ti} . Q_i \quad , \quad (t=G+1,\ldots,N) \tag{5-13}$$

şeklinde tanımlanabilir. Bu eşitlikte;

Q_i : Ana kollardaki akım miktarları sütun vektörü,

d_{ti}: Tali kollara ait göz alt matrisinin transpozesidir.

(5-13) ifadesi matris biçiminde yazılırsa,

$$Q_t = D_{it}^{T} . Q_i \tag{5-14}$$

durumuna gelir ve tali kollardaki akımları ana kollardaki akımlar cinsinden tanımlar (Hartman, 1980).

Akımın korunumu kısıtını ifade eden (5-13) eşitliği amaç fonksiyonunda tali kollara ait elemanların yerine yazılırsa doğrusal olmayan programlama probleminin modeli;

$$U = \frac{1}{3} \left[\sum_{i=1}^{G} (R_i . |Q_i^3|) + \sum_{i=G+1}^{N} (R_i . |(\sum_{t=1}^{G} d_{ti} . Q_i)^3|) \right] -$$

$$\sum_{j=1}^{W} (A_j . Q + \frac{1}{2} B_j . Q^2 + \frac{1}{3} C_j . Q^3) \qquad (5\text{-}15)$$

Amaç fonksiyonu : U → Enküçüklenecek,

Kısıtlar : Yok

olarak şekillenir. Böylece problem çok değişkenli, kısıtsız optimizasyon problemi durumuna gelir.

Örnek Şebeke-1 için amaç fonksiyonu;

$$U = \frac{1}{3} \left[R_1 . |Q_1^3| + R_5 . |Q_5^3| + R_6 . |Q_6^3| + R_2 . |(Q_6 - Q_1)^3| + \right.$$

$$\left. R_3 . |(Q_6 - Q_1 - Q_5)^3| + R_4 . |(Q_6 - Q_5)^3| + R_7 . |(Q_6)^3| \right] - \qquad (5\text{-}16)$$

$$(A_1 . Q_6 + \frac{1}{2} B_1 . Q_6^2 + \frac{1}{3} C_1 . Q_6^3)$$

yapısında olacak, göz sayısı kadar olan değişkenler ana kollardaki Q değerlerinden oluşacaktır.

Uygulanan optimizasyon işlemleri sonucunda ulaşılan optimum Q çözüm seti, ana kollardaki hava miktarlarını verecektir. Tali kollardaki hava miktarları ise, Kirchoff I ilkesinin uygulanması ile kolayca belirlenebilecektir.

5.5. Amaç Fonksiyonunun Analizi

Bir problemin matematiksel modeli oluşturulduktan sonra, bu modele uygulanacak doğrusal olmayan programlama tekniklerinin seçiminde, amaç fonksiyonunun özelliklerinin önemi büyüktür.

Amaç fonksiyonunun konveks veya konkav gibi özel tip bir fonksiyon olması kullanılacak çözüm tekniğini belirler, ulaşılan çözümün yeter şartını sağlar ve çözüm işlemlerinde büyük kolaylıklar getirir.

Havalandırma şebekelerinin modellenmesi ile oluşturulan kısıtsız modelin (5-15) eşitliği ile ifade edilen amaç fonksiyonu, önceki bölümlerde verilen teorik kavramlar doğrultusunda analize sokulmuş ve aşağıda sıralanan özellikleri belirlenmiştir:

a) Tam konveks bir fonksiyondur,
b) Sürekli bir fonksiyondur,
c) Birinci kısmi türevleri vardır ve her nokta için gradyan hesaplanabilir,
d) İkinci kısmi türevleri alınabilir ve Hessien matrisi her nokta için hesaplanabilir,
e) Ayrılabilir bir fonksiyondur,
f) Değişken sayısı şebekeden seçilecek minimum göz sayısı kadar olan çok değişkenli, kısıtsız, üçüncü dereceden bir fonksiyondur.

6. HAVALANDIRMA ŞEBEKELERİNİN KISITSIZ OPTİMİZASYON TEKNİKLERİ İLE ANALİZİ

6.1. Giriş

Havalandırma şebekelerinin modellenmesi ile oluşturulan kısıtsız modele, literatürde güçlü ve hızlı olarak nitelenen ve teorileri önceki bölümlerde verilen kısıtsız optimizasyon teknikleri uygulanmıştır.

Modelin amaç fonksiyonu tam konveks bir fonksiyon olduğundan, gradyanı sıfır olan nokta optimallik gerek ve yeter şartını gerçekleyecek, amaç fonksiyonu bu noktada bütünsel enküçük değerini alacaktır. Ulaşılan optimum çözüm seti havalandırma şebekesindeki toplam güç tüketimini enküçükleyecek hava dağılımını verecektir. Amaç fonksiyonunun değişkenleri ana kollardaki hava miktarları olduğundan, optimum çözüm seti değerleri bu değişkenleri ifade edecektir. Tali kollardaki hava miktarları ise sisteme akımın korunumu ilkesinin uygulanması ile kolayca hesaplanabilecektir.

Oluşturulan modelin ve ele alınan tekniklerin işlerliklerini sınamak için EK-1, EK-2, EK-3 ve EK-4'de verilen şebekeler üzerinde uygulamalar yapılmıştır. Model, elle yapılan işlemler sonucunda, kısıtsız optimizasyon tekniklerini kullanarak örnek şebekeler için çözümlenmiş ve optimum hava miktarı değerleri hesaplanmıştır. Elde edilen sonuçlar aynı şebekelerin Hardy Cross tekniği ile analizi sonucunda bulunan değerlerle karşılaştırılarak, modelin kısıtsız optimizasyon teknikleri ile çözülebilirliği kanıtlanmıştır.

Küçük şebekelerin ele alınan tekniklerle çözümlenmesinde iterasyon işlemlerinin elle yapılması mümkün olabilmekle birlikte, şebekedeki

kol sayısının, dolayısıyla modele giren değişken sayısının fazla olması durumunda sayısal bilgisayarlardan yararlanmak zorunlu olmaktadır. Bir başka deyişle, matematiksel tekniklerin büyük havalandırma şebekelerine uygulanması ancak bilgisayarların kullanılması ile olasıdır. Böylece uzun ve sıkıcı işlemlerden kurtulunmakta, hata yapma olasılığı da ortadan kaldırılmaktadır.

Havalandırma şebekelerinin genellikle fazla olarak nitelendirilebilecek sayıda kol içermesinden dolayı, havalandırma şebekelerine uygulanan doğrusal olmayan programlama teknikleri için bilgisayar programları yazılmıştır. Eldeki bilgisayar olanakları ve çalışma pratikliği açısından bilgisayar kodlamasında BASIC programlama dili kullanılmış, programlar uygulama yapılan örnek şebekeler için çalıştırılarak işlerlikleri kanıtlanmıştır. Çalışmalarda Anadolu Üniversitesi, Mühendislik-Mimarlık Fakültesi Bilgisayar Merkezindeki 128 kB kapasiteli Monroe 8800 tip bilgisayar kullanılmıştır.

Havalandırma şebekelerine has özelliklerden dolayı kısıtsız optimizasyon teknikleri için literatürde verilen bilgisayar programları kullanılmamış, sadece bazı alt problemler için literatürden yararlanma yoluna gidilmiştir.

Bu çalışmada kısıtsız optimizasyon tekniklerinin havalandırma şebekelerinin çözümlenmesindeki çalışma performanslarının Hardy Cross yöntemi ile karşılaştırılması amaçlandığından, göz seçimi ve vantilatör karakteristik eğrisi katsayılarının hesaplanması alt problemleri programlara sokulmamış, doğal havalandırmanın etkisi ise ihmal edilmiştir. Göz dizileri ve vantilatör katsayıları hazır veri olarak okutulmuştur.

Modele uygulanan kısıtsız optimizasyon teknikleri için yazılan bilgisayar programları Ek'lerde verilmiştir.

6.2. En Hızlı İniş Yöntemi

6.2.1. Algoritma

En Hızlı İniş yönteminin, havalandırma şebekeleri için oluşturulan ve (5-15) ifadesi ile verilen kısıtsız modele uygulanmasında, yöntemin genel algoritması izlenmekle birlikte, şebeke analizine özgü bazı değişiklikler yapılmıştır.

1. Adım : Q^0 başlangıç noktası ve istenen hassasiyet değeri (ε) seçilir, k=0 olarak alınır.

2. Adım : $\nabla f(Q_i^k)$ hesaplanır (i=1,2,...,G),
$P_i^k = -\nabla f(Q_i^k)$ olarak atanır.

3. Adım : Tüm kollar için P_j^k değerleri hesaplanır (j=1,2,..,N).

4. Adım : Optimal adım boyutu r^k,
$\min f(|Q_j^k + r^k . P_j^k|)$
olacak şekilde belirlenir. Bu işlem için doğrusal arama yöntemleri kullanılır.

5. Adım : P_i yönünde ilerlenerek ardışık Q noktası,
$Q_j^{k+1} = Q_j^k + r^k . P_j^k$
ifadesinden hesaplanır.

6. Adım : k = k + 1 yapılır, ana kollardaki Q değerleri için,
$|Q_i^k - Q_i^{k-1}| < \varepsilon$
ise işlem bitirilir. Tersi durumda 2. adıma dönülür. İşlem bitirildiğinde ulaşılan Q^k çözüm seti, şebekedeki güç tüketimini enküçükleyen hava dağılımını verecektir.

6.2.2. Başlangıç mümkün çözümü

Amaç fonksiyonu değişkenleri ana kollardaki Q değerleri olduğundan, bunlar için atanacak başlangıç değerleri rassal olarak atanabilir. Tali kollardaki Q değerleri ise, akımın korunumu ilkesini gerçekleyecek şekilde, göz dizilerini kullanarak hesaplanabilir.

Literatürde, başlangıç mümkün çözümünün optimal çözüme yaklaşması durumunda iterasyon sayısının azalacağı rapor edilmektedir (Gottfried and Weisman, 1973 ; Kowalik and Osborne, 1968).

Havalandırma şebekelerinin En Hızlı İniş yöntemiyle çözümlenmesinde çeşitli başlangıç mümkün çözümleri test edilmiş, optimum noktaya çok yakın olma durumu dışında, diğer başlangıç değerlerinin sıfır mümkün çözümüne oranla çok büyük iyileştirmeler yaratmadığı belirlenmiştir. Bazı durumlarda da iterasyon sayısını artırabileceği sonucuna varılmıştır. Bu nedenle, En Hızlı İniş yönteminde ve Newton yöntemi dışında, ele alınmış olan tüm gradyan yöntemlerinde Q° değerleri sıfır olarak atanmıştır. Böylece başlangıç noktası için uygun değerlerin atanması sorunundan da kurtulunmuştur.

6.2.3. Gradyanın hesaplanması

Amaç fonksiyonunun topolojisini değerlendiren kısıtsız optimizasyon tekniklerinde, amaç fonksiyonunun bulunulan noktadaki gradyanının hesaplanması gerekir. Bu nedenle, Q° başlangıç noktasından başlayarak, ulaşılan her ardışık Q^k noktasındaki gradyan belirlenmelidir.

Q=Q' noktasındaki gradyan matematiksel olarak,

$$\nabla f(Q') = \left[\frac{\partial f}{\partial Q_1}, \frac{\partial f}{\partial Q_2}, \ldots\ldots, \frac{\partial f}{\partial Q_G} \right] \qquad (6\text{-}1)$$

vektörü ile ifade edilir.

Bulunulan noktadaki gradyanın hesaplanabilmesi için, amaç fonksiyonunun Q_i , (i=1,2,..,G) değişkenlerine göre kısmi türevlerinin alınıp, Q' mümkün çözüm setinin elde edilen denklem sisteminde yerine konulması gerekir.

Örnek Şebeke-1 için (5-16) eşitliği ile verilen amaç fonksiyonunun Q_i , (i=1,2,..,G) değişkenlerine göre birinci kısmi türevleri alınıp sıfıra eşitlenirse;

$$\frac{\partial U}{\partial Q_1} = R_1.|Q_1|.Q_1 - R_2.|Q_6-Q_1|.(Q_6-Q_1) - R_3.|Q_6-Q_1-Q_5|.(Q_6-Q_1-Q_5) = 0$$

$$\frac{\partial U}{\partial Q_5} = R_5.|Q_5|.Q_5 - R_3.|Q_6-Q_1-Q_5|.(Q_6-Q_1-Q_5) - R_4.|Q_6-Q_5|.(Q_6-Q_5) = 0$$

$$\frac{\partial U}{\partial Q_6} = R_6.|Q_6|.Q_6 + R_2.|Q_6-Q_1|.(Q_6-Q_1) + R_3.|Q_6-Q_1-Q_5|.(Q_6-Q_1-Q_5) +$$

$$R_4.|Q_6-Q_5|.(Q_6-Q_5) + R_7.|Q_6|.Q_6 - (A_1+B_1.Q_6+C_1.Q_6^2) = 0 \qquad (6\text{-}2)$$

denklem sistemi elde edilir. Optimum çözüme ulaşıldığında gradyan vektörü,

$$\nabla U = \left[\frac{\partial U}{\partial Q_1} ; \frac{\partial U}{\partial Q_5} ; \frac{\partial U}{\partial Q_6} \right] \simeq 0 \qquad (6\text{-}3)$$

olacaktır.

Kolayca gözlemlenebileceği gibi, kısmi türevleri tanımlayan (6-2) denklem sistemi havalandırma şebeke teorisinde Kirchoff II yasası olarak bilinen, gözler etrafındaki basınç düşüşlerinin korunumu ilkesini gerçeklemektedir. Örnek Şebeke-1'e bu ilke uygulanacak olursa (6-2) denklem sistemi elde edilir. Optimum çözüm setine ulaşıldığında bu denklem sistemi sağlanmış olacak, başka bir deyişle hesaplanmış Q_i değerleri gözlerdeki basınç düşüşleri toplamlarını sıfır yapacaktır. Örnek Şebeke-1 için,

$Q^o = (0 ; 0 ; 0)$

vektörü başlangıç mümkün çözümü olarak alındığında gradyan vektörü,

$\nabla U = (0 ; 0 ; -494,54)$

olarak hesaplanacaktır. Gradyan vektörünün negatifi olan,

$P^1 = (0 ; 0 ; 494,54)$

vektörü yönünde ilerlenerek ardışık Q^1 noktasına ulaşılacaktır.

(6-2) denklem sistemi Kirchoff II yasasını gerçeklediğinden, amaç fonksiyonunun bulunulan noktadaki gradyanının hesaplanmasında, sayısal türev alma yöntemlerine başvurulmaksızın, göz dizilerinden yararlanılmıştır. Her gözdeki kol dizileri sırasıyla bilgisayarda taratılarak, gradyan vektörü sayısal olarak oluşturulmuştur.

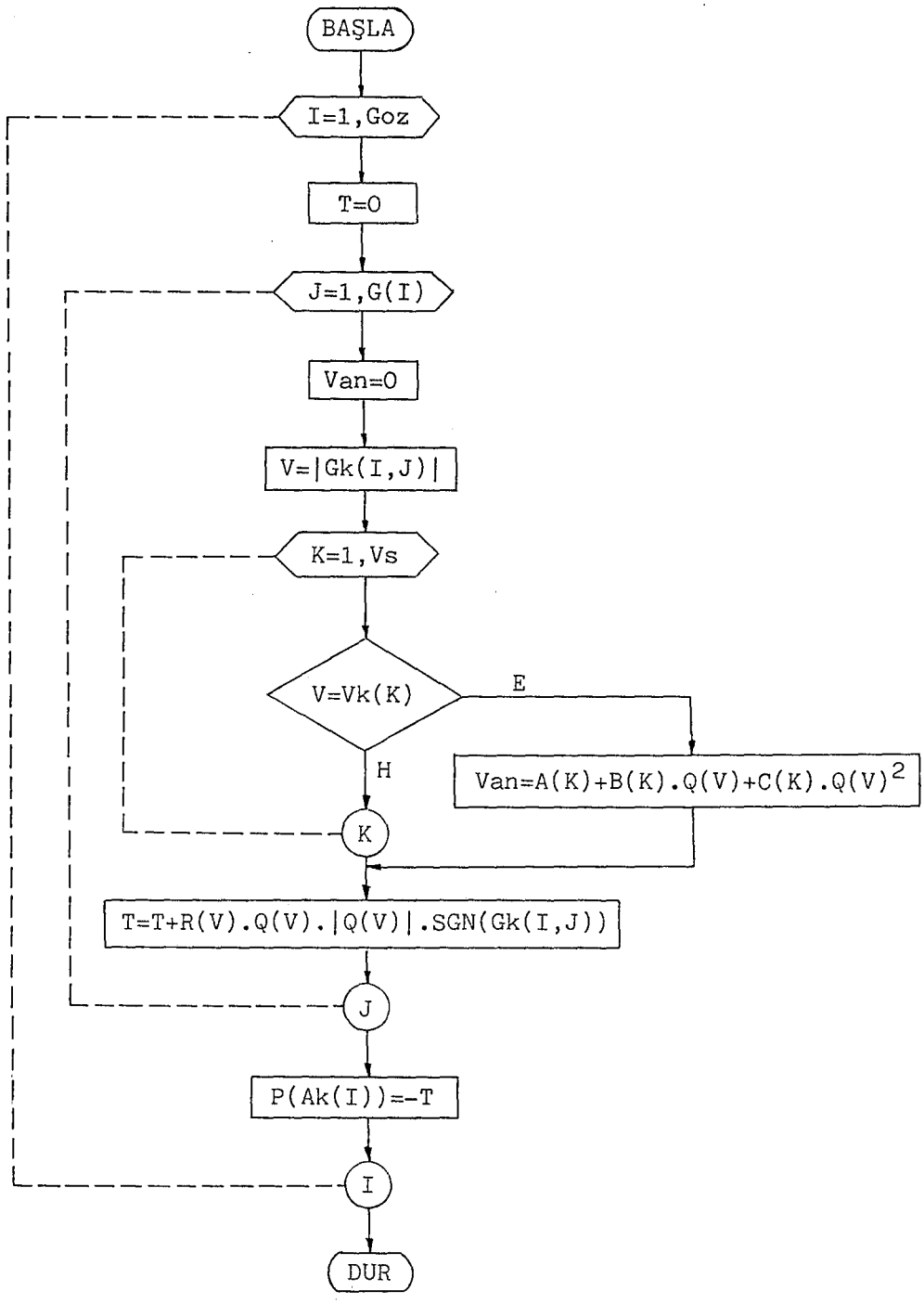
Bulunulan noktadaki gradyan vektörünün hesaplanması için yazılan ve ele alınan kısıtsız optimizasyon tekniklerinin bilgisayar programlarında alt program olarak kullanılan, gradyan alt programının akış şeması Şekil 6,1'de verilmiştir.

6.2.4. Tek boyutlu optimizasyon

Amaç fonksiyonunun bulunulan noktadaki gradyanı hesaplandıktan sonra, algoritmanın 4. adımı işlenecektir. Matematiksel işlemlerin en güç ve zaman alıcı bölümü, bu adımdaki optimal adım boyutunun saptanması işlemidir. Burada problem,

$$\min f[||(Q^k - r^k.\nabla f(Q^k))||] \qquad (6\text{-}4)$$

koşulunu gerçekleyen r^k değerinin bulunmasıdır. Q^k ve $\nabla f(Q^k)$ parametreleri atanmış veya hesaplanmış sayısal değerler olduğundan problem, r^k değişkeninin hesaplanmasına yönelik tek boyutlu arama problemi olarak şekillenir.



Şekil 6.1 : Gradyan hesaplama alt programının akış şeması.

Örnek Şebeke-1 için gradyan hesaplandıktan sonra ilk iterasyonda,

$$U = \frac{1}{3}\,[\,0{,}1.|(0-r^{\circ}.0)^3| + 0{,}9.|(0-r^{\circ}.0)^3| + 2{,}5.|(0-r^{\circ}.(-494{,}54))^3| +$$

$$0{,}4.|(0-r^{\circ}.(-494{,}54))^3| + 0{,}08.|(0-r^{\circ}.(-494{,}54))^3| +$$

$$0{,}6.|(0-r^{\circ}.(-494{,}54))^3| + 0{,}5.|(0-r^{\circ}.(-494{,}54))^3|\,] -$$

$$494{,}54.(0-494{,}54) + \frac{6{,}9}{2}.(0-494{,}54)^2 - \frac{0{,}047}{3}.(0-494{,}54)^3$$

tek değişkenli fonksiyonunu enküçükleyen r° değerinin hesaplanması gerekir.

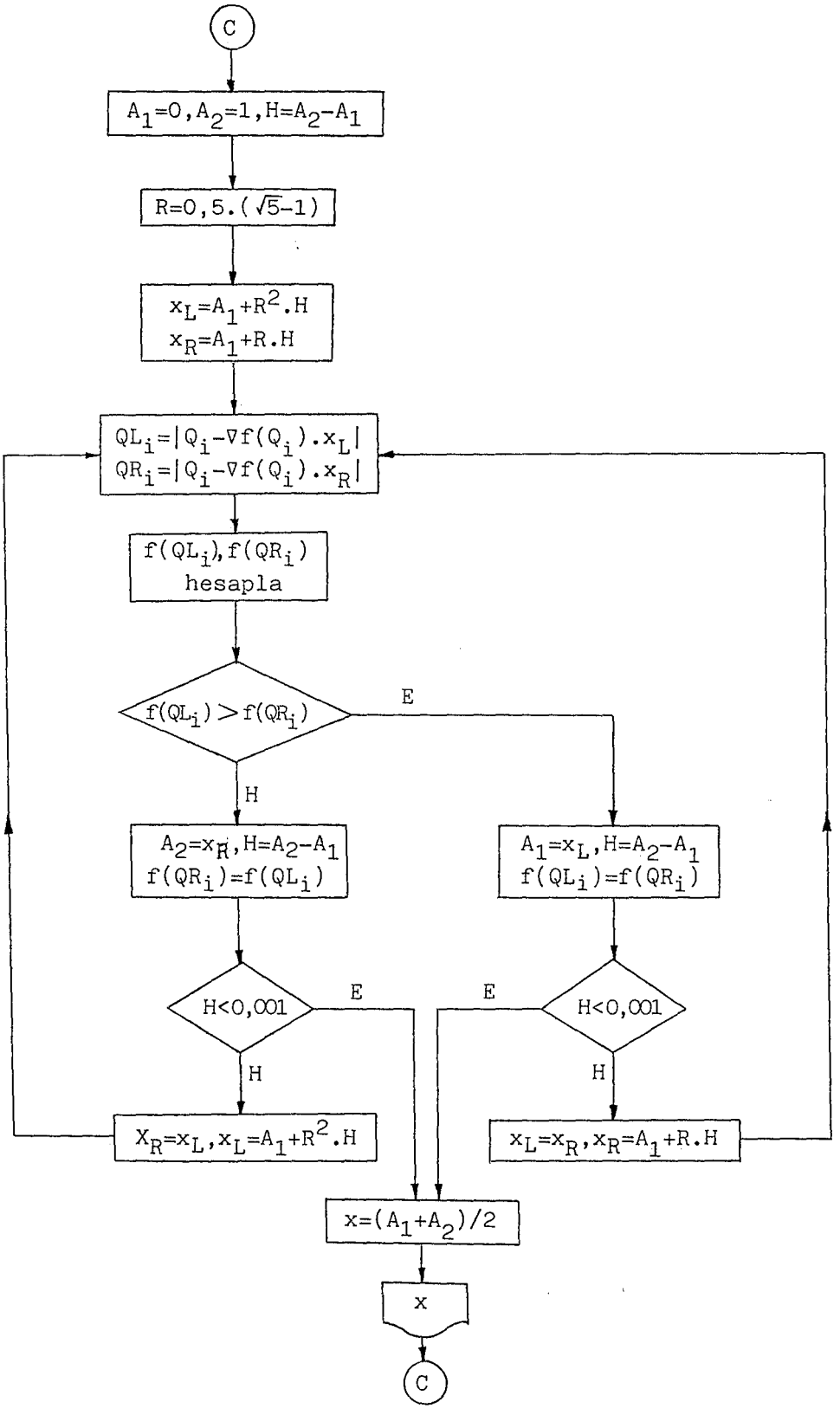
Amaç fonksiyonunun tek modlu olmasını sağlamak için mutlak değerlerin kullanılmış olması, r° değişkenine göre türev alınmasıyla oluşan ikinci dereceden denklemin pozitif kökünün alınmasını engellemektedir. Bu durumda, doğrusal arama tekniklerinden birinin kullanılması gerekecektir.

Literatürde en güçlü ve hızlı tek boyutlu arama teknikleri olarak Altın Oran, Fibonacci ve Kareli İnterpolasyon yöntemleri anılmaktadır (Wilde, 1964 ; Simmons, 1975 ; Avriel, 1976).

Fibonacci yöntemi, tekrarlama sayısının önceden belirlenmesi sakıncasından dolayı uygulama dışı bırakılmış, Altın Oran ve Kareli İnterpolasyon yöntemleri probleme uygulanarak, çalışma hızları karşılaştırılmıştır.

Altın Oran yönteminin uygulanmasında 3.9.4. bölümünde verilen algoritma aynen izlenmiş, (0-1) aralığında doğrusal arama yapılarak 0,001 duyarlılıkla optimal adım boyutu belirlenmiştir. Yöntemin algoritması bilgisayara kodlanarak bir alt program yazılmış, her iterasyonda ana programdan bu alt programa gidilerek gerekli r^k değerinin alınması sağlanmıştır.

Altın Oran yöntemi için yazılan alt programın genelleştirilmiş akış şeması Şekil 6,2'de verilmiştir. Tek boyutlu aramada bu tekniği kulla-



Şekil 6.2 : Altın Oran tekniği alt programının akış şeması.

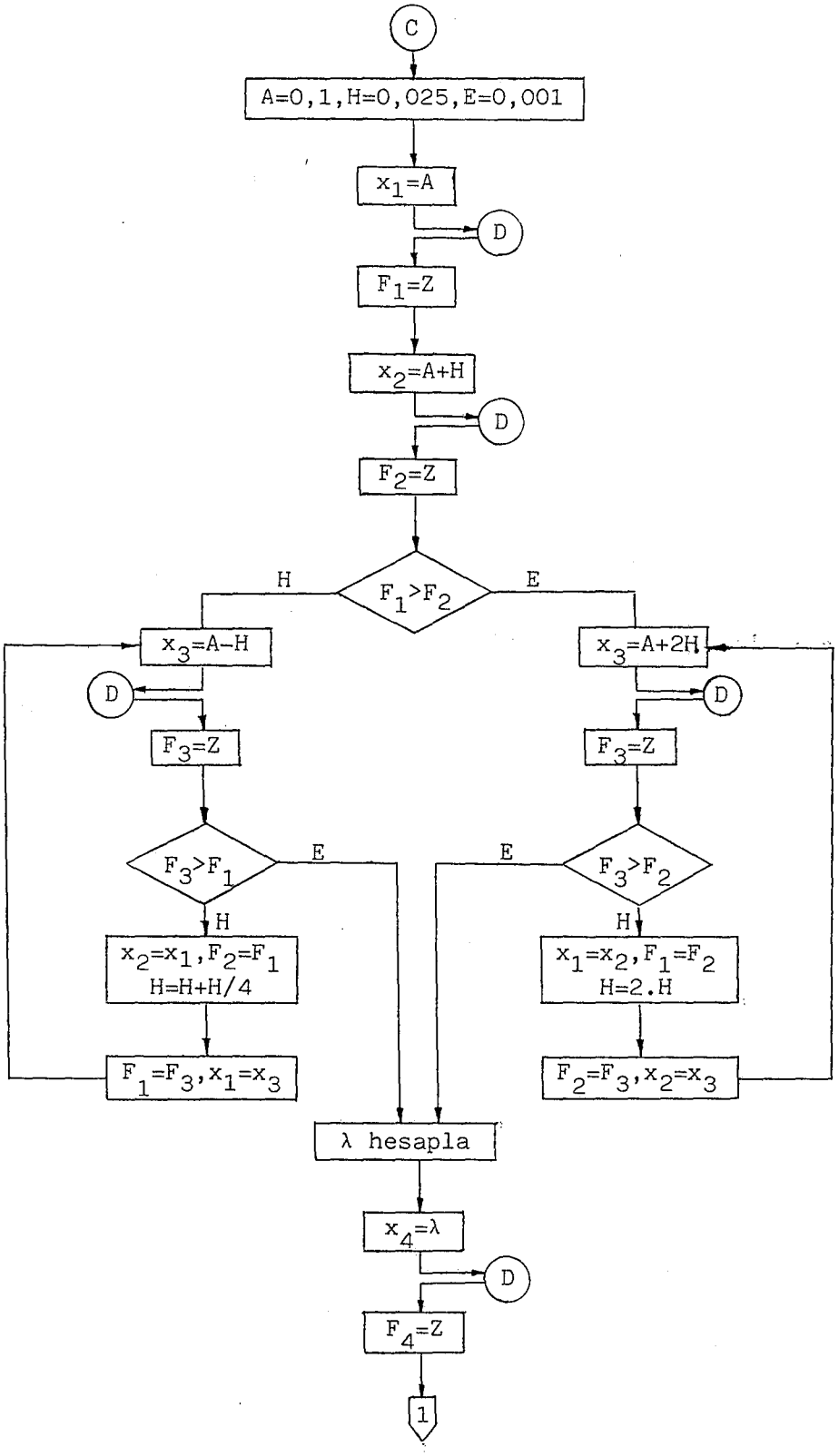
nan En Hızlı İniş yönteminin bilgisayar programı ise EK-6'da verilmiştir.

Kareli İnterpolasyon tekniğinin uygulanmasında, 3.9.5. bölümünde verilen algoritma izlenmekle birlikte, yönteme daha hızlı işlerlik kazandırmak için algoritmaya bazı katkılar yapılmıştır. Bunday (1984) tarafından verilen algoritmada ikinci adımdan hemen sonra interpolasyona geçilmektedir. Bazı durumlarda, ilk iki adımda ele alınan üç noktadaki fonksiyon değerlerinden en küçük olanı arada kalmamaktadır. Bu düzenle hemen interpolasyona geçilmesi durumunda, iterasyon sayısı ve hesaplama süresi fazla olmaktadır. Bu sakıncayı gidermek için ikinci adımdan sonra, minimum nokta arada kalana kadar aralığı daraltma yoluna gidilmiştir. İkinci ve daha sonraki tekrarlamalarda interpolasyon işlemi, (3-33) ifadesi kullanılarak yapılmıştır. Bu düzenlemelerle yöntemin çok daha hızlı çalıştığı belirlenmiştir.

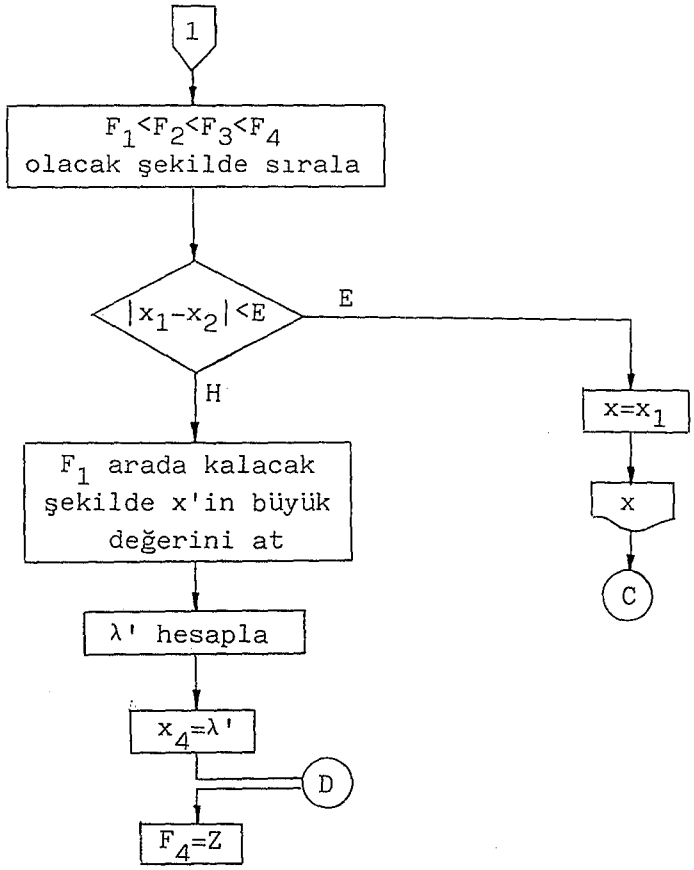
Kareli İnterpolasyon yöntemi için bir alt program yazılarak, doğrusal aramada bu yöntemi kullanan kısıtsız optimizasyon tekniklerinde, ana programdan bu alt programa gidilerek optimal adım boyutunun belirlenmesi sağlanmıştır. r^k değerinin genellikle (0-0,1) arasında kaldığı gözlemlendiğinden, başlangıç noktası 0,05 ve adım uzunluğu 0,025 olarak alınmış, hesaplamalar 0,001 duyarlılıkla yapılmıştır.

Kareli İnterpolasyon alt programının genelleştirilmiş akış şeması Şekil 6,3'de, doğrusal aramada bu tekniği kullanan En Hızlı İniş yönteminin bilgisayar programı ise EK-7'de verilmektedir.

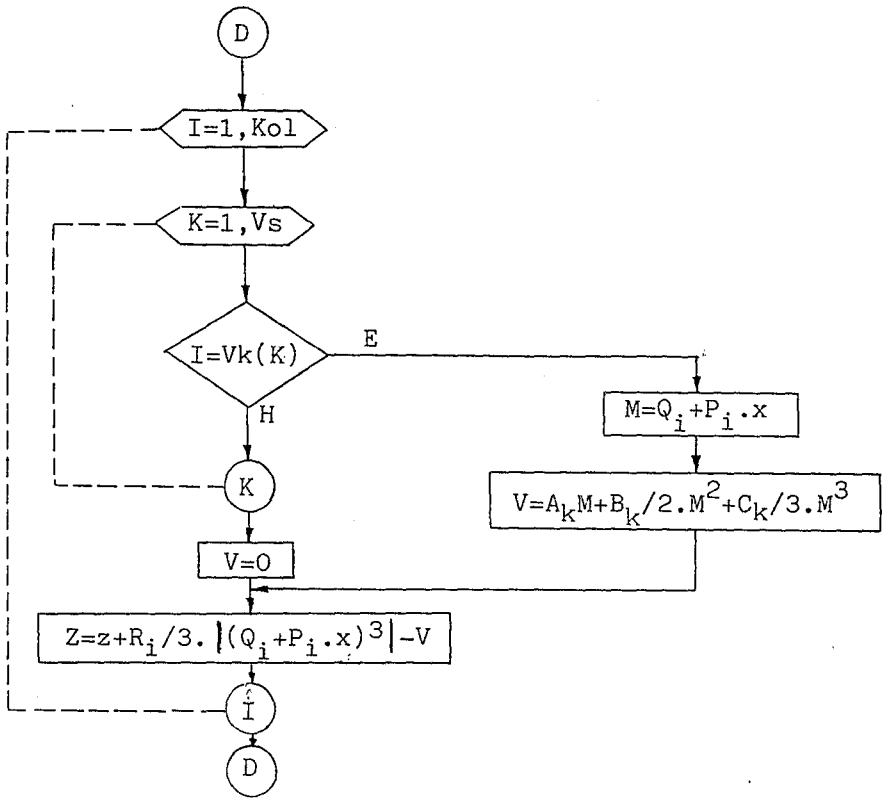
Kareli İnterpolasyon ve Altın Oran tek boyutlu arama yöntemlerinin çalışma hızlarını karşılaştırmak amacıyla, bu teknikleri kullanan En Hızlı İniş yönteminin bilgisayar programları örnek şebekeler için çalıştırılmıştır. Karşılaştırma ölçütü olarak çözümde harcanan toplam süre, iterasyon sayısı ve amaç fonksiyonu değerinin hesaplanma sayısı alınmıştır. Elde edilen sonuçlar Çizelge 6,1'de verilmektedir.



Şekil 6.3 : Kareli interpolasyon alt programı akış şeması.



Şekil 6.3 : (devam)



Şekil 6.4 : Fonksiyon değerini hesaplama alt programı akış şeması.

Çizelge 6.1 : Altın Oran ve Kareli İnterpolasyon yöntemlerinin karşılaştırılması.

		Kareli İnterpolasyon	Altın Oran
Şebeke-1	İterasyon Sayısı	8	8
	Çözüm Süresi (s)	21	89
	Fonksiyon Hesaplama Sayısı	51	896
Şebeke-3	İterasyon Sayısı	19	19
	Çözüm Süresi (s)	93	424
	Fonksiyon Hesaplama Sayısı	137	3952

Çizelge sonuçlarından görülebileceği gibi, Altın Oran yönteminde amaç fonksiyonu değerinin hesaplanma sayısı çok daha fazla olmakta, dolayısıyla çözüm süresi kareli interpolasyon yapılmasına oranla büyük bir artış göstermektedir. Şebekedeki kol sayısı arttıkça bu fark daha da belirginleşmektedir. Bu gözlem literatür ile de uyum göstermektedir (Wilde, 1964 ; Himmelblau, 1972).

Kareli İnterpolasyon yönteminin, elde edilebilen en hızlı sonucu vermesinden dolayı, tek boyutlu arama yapılan tüm kısıtsız optimizasyon tekniklerinde bu yöntem kullanılmıştır.

6.2.5. İterasyon bitirme ölçütü

Doğrusal arama alt programında r^k değeri belirlendikten sonra, algoritmanın beşinci adımında ardışık Q^{k+1} noktası saptanır. Altıncı adımda ise ulaşılan noktanın optimalliği test edilir.

Optimum nokta için literatürde, gradyanın belli bir yakınlıkla sıfır olması koşulu verilmektedir (Polak, 1971). Havalandırma şebekelerinin analizinde, Hardy Cross yöntemindekine uyum sağlamak için, iterasyon

işlemlerini bitirme ölçütü olarak, ulaşılan nokta ile bir önceki nokta arasındaki farkın, önceden belirlenen bir hassasiyet değerinden az olması koşulu kullanılmıştır. Altıncı adımda,

$$|Q_i^{k+1} - Q_i^k| < \varepsilon \quad , \quad (i=1,2,..,G) \qquad (6\text{-}5)$$

koşulu gerçeklenirse işlem bitirilmekte, tersi durumda ikinci adıma dönülmektedir.

Uygulamalarda ε=0,005 m^3/s olarak alınmış, amaç fonksiyonunun değişkenlerini oluşturan ana kollardaki hava miktarları optimum çözüm seti olarak, bu duyarlılıkla hesaplanmıştır. Tali kollardaki hava miktarları ise, Kirchoff I ilkesinin sisteme uygulanması ile belirlenmiştir.

6.2.6. Bilgisayar programı

En Hızlı İniş yönteminin algoritması BASIC programlama dilinde kodlanarak, örnek şebekeler bilgisayar yardımıyla çözümlenmiştir. Bilgisayar kodlamasında kullanılan temel sabit ve değişkenlerin isim ve boyutları aşağıdaki gibidir:

Kol : Şebekedeki kol sayısı,
Goz : Göz sayısı,
Vs : Şebekedeki vantilatör sayısı,
G(I) : I. gözdeki kol sayısı, G(Goz),
Gk(I,J) : I. gözdeki kol numaraları, Gk(Goz,Kol),
Ak(I) : I. gözdeki ana kolun numarası, Ak(Goz),
R(I) : Kol dirençleri, R(Kol),
Q(I) : Kollardaki hava miktarları, Q(Kol),
A(I) : I. vantilatörün A katsayısı, A(Vs),
B(I) : I. vantilatörün B katsayısı, B(Vs),
C(I) : I. vantilatörün C katsayısı, C(Vs),
Vk(I) : I. vantilatörün bulunduğu kol numarası, Vk(Vs),
P(I) : Negatif gradyan vektörü, P(Kol),
x : Optimal adım boyutu,
Z9 : Amaç fonksiyonunun değeri,
It : İterasyon sayısı,
A9 : Kareli interpolasyonda başlangıç noktası,
H9 : Kareli interpolasyonda adım uzunluğu,

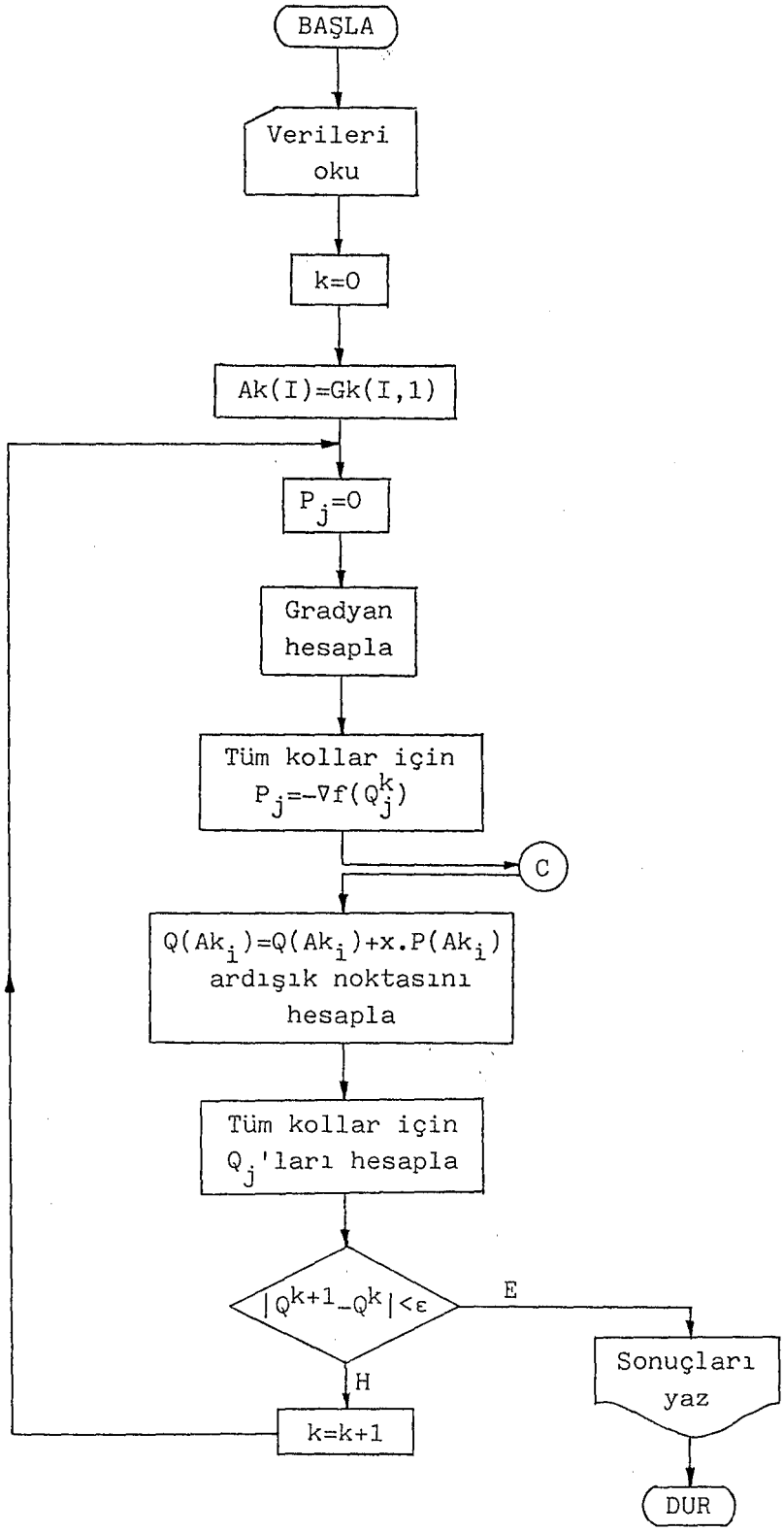
En Hızlı İniş yöntemi için, doğrusal arama alt programında birisi Altın Oran yöntemini, diğeri Kareli İnterpolasyon yöntemini kullanan, iki bilgisayar programı yazılmış, ikinci programın daha hızlı çalıştığı belirlenmiştir. Her iki programda ana programlar aynı olup, bu programın genelleştirilmiş akış şeması Şekil 6,5'de verilmektedir. Doğrusal aramada Altın Oran yöntemini kullanan program EK-6'da, kareli interpolasyon yapan ve daha hızlı çalıştığı için karşılaştırmalarda temel alınan program EK-7'dedir.

Programa veri olarak sırasıyla şebekedeki kol sayısı, göz sayısı, vantilatör sayısı, vantilatörlerin katsayıları, katsayıları verilen vantilatörün bulunduğu kol numarası, kilomurgue olarak kolların direnç değerleri, her gözde bulunan kol sayısı ve o gözdeki kol numaraları verilmekte, çıktı olarak her koldaki hava miktarları (m^3/s) olarak alınmaktadır. Örnek Şebeke-1 için verilerin okutma düzeni EK-7'deki programın veri satırlarındaki gibi olmuştur. Bu düzen Newton yöntemi dışında, bundan sonra verilen tüm bilgisayar programları için de aynen kullanılmıştır.

6.2.7. Optimum çözüm seti

Optimum çözüm seti, amaç fonksiyonunu enküçükleyen karar değişkenleri değerlerinden oluşmaktadır. Başka bir deyişle optimum çözüm seti, havalandırma şebekesindeki güç tüketimini minimum yapan, ana kollardaki hava miktarlarını ifade etmektedir. Tali kollardaki hava miktarları ise, akımın korunumu ilkesini göz dizilerini kullanarak uygulamak suretiyle bilgisayarda hesaplatılmaktadır.

En Hızlı İniş yönteminin bilgisayar programı örnek şebekeler için çalıştırılarak optimum çözüm setleri çıktı olarak alınmıştır. Örnek Şebeke-1 için yöntemin her iterasyondaki işleyişi Çizelge 6,2'deki gibi olmuştur. Çizelge 6,3'de ise, örnek şebekeler için ulaşılan op-



Şekil 6.5 : En Hızlı İniş yöntemi bilgisayar programının genelleştirilmiş akış şeması.

Çizelge 6.2 : Örnek Şebeke-1 için En Hızlı İniş yönteminin işleyişi.

İterasyon (k)		0	1	2	3	4	5	6	7	8
Q^k	1	0	0	3,6214	3,6893	4,3063	4,3131	4,3801	4,3799	4,3831
	2	0	10,2189	6,8056	7,5317	6,8678	6,9458	6,8754	6,8834	6,8835
	3	0	10,2189	1,6753	2,4812	1,7926	1,8778	1,7963	1,8051	1,8049
	4	0	10,2189	5,2967	6,1705	6,0989	6,1908	6,1765	6,1851	6,1879
	5	0	0	5,1303	5,0505	5,0751	5,0681	5,0791	5,0780	5,0786
	6	0	10,2189	10,4271	11,2211	11,1738	11,2588	11,2556	11,2633	11,2666
	7	0	10,2189	10,4271	11,2211	11,1738	11,2588	11,2556	11,2633	11,2666
$-\nabla f(Q^k)$	1	0	50,124	5,6369	9,5729	0,5789	0,9774	-0,0186	0,0297	
	2	494.54	-47,242	60,3108	-10,302	6,7127	-1,0253	0,7312	0,0007	
	3	494,54	-118,252	66,9409	-10,683	7,3179	-1,1869	0,8015	-0,0021	
	4	494,54	-68,128	72,5778	-1,110	7,8969	-0,2094	0,7829	0,0276	
	5	0	71,009	-6,6301	0,381	-0,605	0,1616	-0,0703	0,0028	
	6	494,54	2,881	65,9478	-0,729	7,2917	-0,0478	0,7126	0,0305	
	7	494,54	2,881	65,9478	-0,729	7,2917	-0,0478	0,7126	0,0305	
r^k		0,0207	0,072	0,0121	0,064	0,0116	0,0686	0,0109	0,1075	

Çizelge 6.3 : En Hızlı İniş yöntemiyle elde edilen çözümlerin, Hardy Cross tekniği sonuçlarıyla karşılaştırılması.

Örnek Şebeke	Kol	En Hızlı İniş	Hardy Cross	Mutlak Fark	S_q
1	1	4,3831	4,3822	0,0009	0,0019
	2	6,8835	6,8818	0,0017	
	3	1,8049	1,8031	0,0018	
	4	6,1879	6,1853	0,0026	
	5	5,0786	5,0787	0,0001	
	6	11,2666	11,2641	0,0025	
	7	11,2666	11,2641	0,0025	
2	1	2,3738	2,3731	0,0007	0,0126
	2	8,2571	8,2591	0,0020	
	3	4,8071	4,7882	0,0189	
	4	7,1809	7,1613	0,0196	
	5	6,3819	6,3822	0,0003	
	6	0,7991	0,7791	0,0201	
	7	3,4498	3,4708	0,0210	
	8	4,2488	4,2501	0,0013	
	9	10,6308	10,6322	0,0014	
	10	10,6308	10,6322	0,0014	
3	1	9,8111	9,8101	0,0010	0,0054
	2	4,3489	4,3491	0,0002	
	3	4,6281	4,6314	0,0033	
	4	4,2961	4,2896	0,0061	
	5	5,4621	5,4611	0,0010	
	6	0,2791	0,2824	0,0033	
	7	-0,3319	-0,3418	0,0099	
	8	5,5149	5,5204	0,0055	
	9	3,3253	3,3223	0,0030	
	10	3,0462	3,0399	0,0063	
	11	2,1367	2,1387	0,0020	
	12	2,4687	2,4805	0,0118	
	13	9,8111	9,8101	0,0010	

S_q : Tahminin standard hatası.

timum çözüm setlerinin, Hardy Cross tekniği ile elde edilen sonuçlarla karşılaştırılması verilmektedir. Bu çizelgede, her kol için ortaya çıkan farklar mutlak fark olarak alınmış, tüm şebekedeki farkı ifade etmek için, istatistikte "Tahminin Standard Hatası" olarak bilinen;

$$S_q = \frac{\Sigma(Q_i - Q_i')^2}{N} \quad , \quad (i=1,2,..,N) \qquad (6\text{-}6)$$

eşitliği kullanılmıştır (Gürtan, 1982). Burada;

S_q : Tahminin standard hatası,

Q_i : Hardy Cross tekniği ile hesaplanan hava miktarı,

Q_i' : Ele alınan optimizasyon tekniği ile hesaplanan hava miktarı,

N : Şebekedeki kol sayısıdır.

Çizelge değerlerinden, En Hızlı İniş yönteminin havalandırma şebekelerine uygulanması ile alınan sonuçların yeterince hassas olduğu anlaşılmaktadır. Aradaki küçük fark, ∓0,005 değerindeki hassasiyet miktarından, Hardy Cross tekniğinde (2-11) açınımının ikinci dereceden teriminin ihmal edilmesinden ve matematiksel işlemlerdeki yuvarlatma hatalarından kaynaklanmaktadır. Hassasiyet değerinin daha küçük alınmasının, bu farkı daha da azaltacağı açıktır.

En Hızlı İniş yönteminin örnek şebekelere uygulanması ile alınan sonuçlar, oluşturulan matematiksel modelin bu yöntem ile çözülebilirliğini kanıtlamış olmaktadır.

6.3. Fletcher-Reeves Yöntemi

6.3.1. Algoritma

Havalandırma şebekeleri için oluşturulan kısıtsız modelin, bileşke yönü kullanan Fletcher-Reeves yöntemiyle çözümlenmesinde, şu algoritma izlenmiştir:

1. Adım : Q^0 başlangıç noktası ve istenen hassasiyet değeri (ε) seçilir, k=0, t=1 alınır.

2. Adım : $\nabla f(Q_i^k)$ gradyanı hesaplanır (i=1,2,..,G), tüm kollar için; $P^k = -\nabla f(Q_j^k)$ olarak atanır (j=1,2,..,N).

3. Adım : $r^k = \min f[|(Q_j^k - r^k.P_j^k)|]$ yapan optimal adım boyutu, kareli interpolasyon tekniği ile hesaplanır.

4. Adım : Ardışık Q^{k+1} noktası, $Q_j^{k+1} = Q_j^k + r^k.P^k$ ifadesince belirlenir.

5. Adım : $|Q_i^{k+1} - Q_i^k| < \varepsilon$ ise işlem bitirilir.

6. Adım : t=G ise t=1 yapılıp 2. adıma dönülür, t<G ise $\nabla f(Q_i^{k+1})$ hesaplanır.

7. Adım : $$\beta^k = \frac{\nabla f(Q^{k+1})'.\ \nabla f(Q^{k+1})}{\nabla f(Q^k)'.\ \nabla f(Q^k)}$$

hesaplandıktan sonra bileşke yön,

$$P^{k+1} = -\nabla f(Q^{k+1}) + \beta^k.P^k$$

olarak belirlenir.

8. Adım : k=k+1 yapılıp 3. adıma dönülür.

6.3.2. Bilgisayar programı

Verilen algoritma BASIC programlama dilinde kodlanarak örnek şebekeler bilgisayar yardımıyla çözümlenmiş ve yöntemin havalandırma şebekelerine uygulanabilirliği kanıtlanmıştır.

Program, bir ana ve iki alt programdan oluşmaktadır. Algoritmanın ikinci adımında gradyan alt programına gidilerek, bulunulan nokta-

daki gradyan vektörü hesaplanmaktadır. Belirlenen gradyan vektörünü kullanarak, algoritmanın üçüncü adımında optimal adım boyutu hesaplanmaktadır. Bu işlem için, Kareli İnterpolasyon tekniğini kullanan tek boyutlu arama alt programına gidilmektedir. Her iki alt program da En Hızlı İniş yönteminde kullanıldığı biçimdedir.

Hesaplanan optimal adım boyutunu kullanarak, ilk iterasyonda negatif gradyan vektörü yönünde, sonraki iterasyonlarda ise, hesaplanan bileşke yönde ilerlenerek ardışık noktalar belirlenmektedir.

Fletcher-Reeves yöntemi için yazılan bilgisayar programı EK-8'de, programın mantıksal işleyişini gösteren genelleştirilmiş akış şeması ise Şekil 6,6'da verilmiştir.

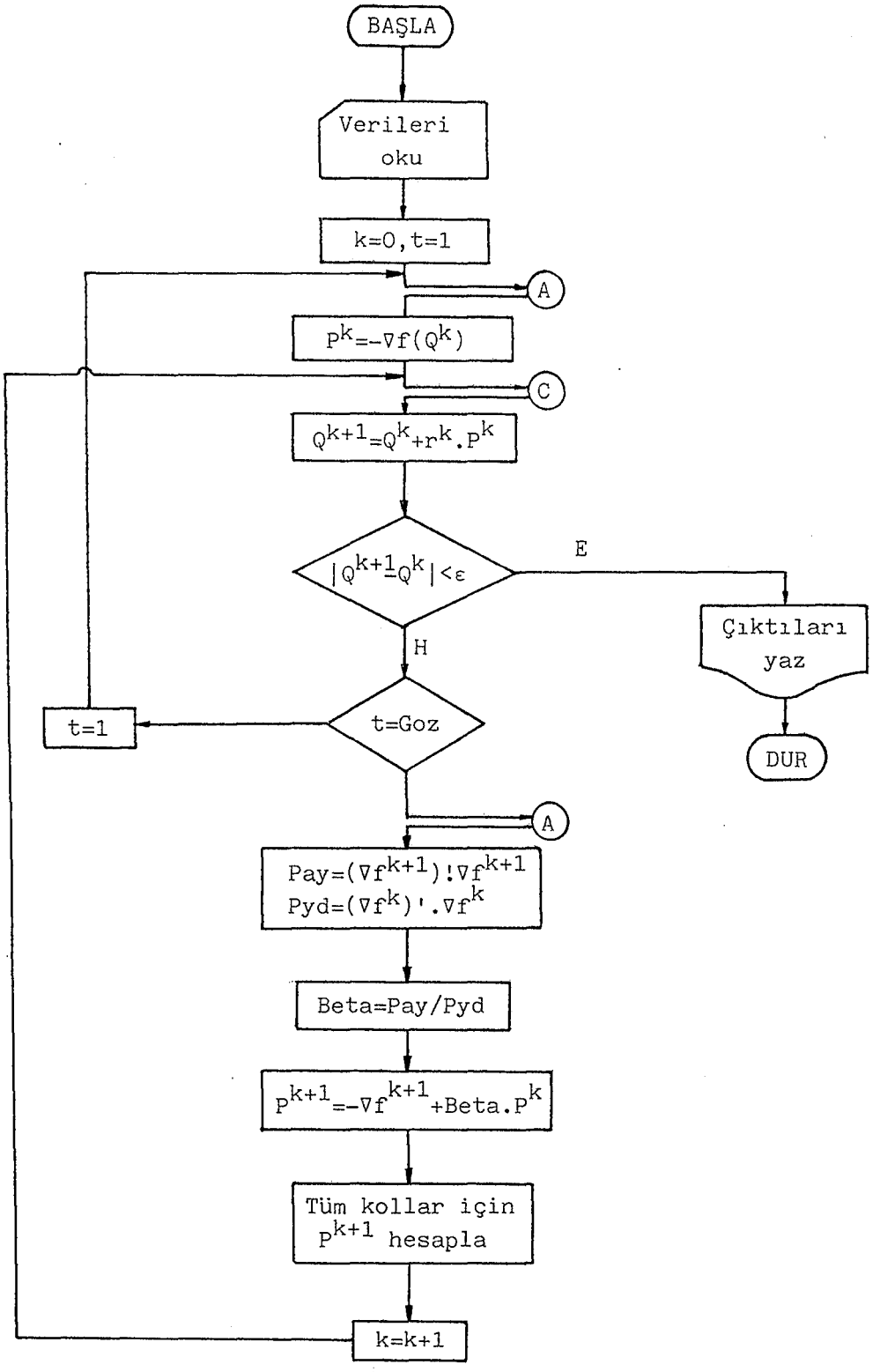
Programda, sabit ve değişkenlerin isimleri, En Hızlı İniş yöntemiyle uyum sağlamak için, bu yöntemde kullanıldığı gibi alınmıştır. Farklı olarak kullanılan sabit ve değişken isimleri ile boyutları aşağıdaki gibi olmuştur:

D(I) : Bir önceki noktadaki gradyan vektörü, D(Kol),
P(I) : Bileşke yön vektörü, P(Kol),
P2(I) : Bir önceki noktadaki bileşke yön vektörü, P2(Kol),
Pay : β eşitliğinin payı,
Pyd : β eşitliğinin paydası,
Beta : β parametresi.

Programa veri okutma düzeni En Hızlı İniş yöntemindeki gibi olup, havalandırma şebekesinin kollarındaki hava dağılımı değerleri çıktı olarak alınmaktadır.

6.3.3. Optimum çözüm seti

Yöntemin uygulanmasında başlangıç Q° değerleri sıfır olarak atanmış, iterasyon işlemleri ulaşılan nokta ile bir önceki nokta arasındaki



Şekil 6.6 : Fletcher-Reeves yöntemi bilgisayar programının genelleştirilmiş akış şeması.

fark, istenen hassasiyet değerinin altında kalana dek sürdürülmüştür.

Fletcher ve Reeves (1964) tarafından önerildiği gibi, her (G+1) iterasyondan sonra ikinci adıma dönülerek, hesaplamaya $-\nabla f(Q)$ yönünde yeniden başlanmıştır. Bu işlemin toplam çözüm süresini gerçekten de azalttığı gözlenmiştir.

Yöntemin Örnek Şebeke-1 için işleyişi Çizelge 6,4'deki gibi olmuştur. Örnek şebekeler için ulaşılan optimum noktaların, Hardy Cross tekniği ile elde edilen sonuçlarla karşılaştırılması ise Çizelge 6,5'de verilmiştir.

Çizelge değerleri, havalandırma şebekelerinin Fletcher-Reeves yöntemi ile çözümlenerek, kollardaki hava dağılımının yeterli bir hassasiyetle hesaplanabileceğine işaret etmektedir. Bu yöntemde bileşke yönler kullanıldığından, En Hızlı İniş yöntemine oranla daha az iterasyonda optimum noktaya ulaşılmış, Hardy Cross yöntemindekine daha yakın sonuçlar alınmıştır.

6.4. Newton Yöntemi

6.4.1. Algoritma

Havalandırma şebekelerinin Newton yöntemi ile analizinde izlenen algoritma aşağıdaki gibi verilebilir:

1. Adım : Q^0 başlangıç noktası ve istenen hassasiyet değeri (ε) seçilir, $k=0$ alınır.

2. Adım : Q^k noktası için Hessien matrisi oluşturulur.

3. Adım : Hessien matrisinin inversi hesaplanır.

4. Adım : $\nabla f(Q_i^k)$ hesaplanır $(i=1,2,..,G)$.

5. Adım : $Q_i^{k+1} = Q_i^k - H(Q_i^k)^{-1} . \nabla f(Q_i^k)$
ardışık noktası belirlenir.

Çizelge 6.4 : Örnek Şebeke-1 için Fletcher-Reeves yönteminin işleyişi.

İterasyon (k)		0	1	2	3	4	5	6	7
Q^k	1	0	0	3,3321	4,0027	4,3309	4,3555	4,3851	4,3851
	2	0	10,1235	8,5756	7,2846	6,8777	6,9045	6,8801	6,8801
	3	0	10,1235	3,8552	1,8079	1,7899	1,8466	1,7997	1,7996
	4	0	10,1235	7,1873	5,8106	6,1208	6,2020	6,1848	6,1847
	5	0	0	4,7204	5,4767	5,0879	5,0580	5,0804	5,0804
	6	0	10,1235	11,9077	11,2873	11,2087	11,2601	11,2652	11,2651
	7	0	10,1235	11,9077	11,2873	11,2087	11,2601	11,2652	11,2651
p^k	1	0	49,1928	53,664	5,4664	1,9941	0,3951	-0,0357	
	2	494,54	-22,8519	-103,309	-6,7761	2,1752	-0,3264	-0,0454	
	3	494,54	-92,5417	-163,834	-0,2997	4,6021	-0,6249	-0,0261	
	4	494,54	-43,3489	-110,17	5,1667	6,5962	-0,2299	-0,0617	
	5	0	69,6898	60,525	-6,4764	-2,4269	0,2986	-0,0194	
	6	494,54	26,3409	-49,644	-1,3097	4,1693	0,0687	-0,0811	
	7	494,54	26,3409	-49,644	-1,3097	4,1693	0,0687	-0,0811	
r^k		0,0205	0,0677	0,0125	0,0601	0,0123	0,075	0,0012	

Çizelge 6.5 : Fletcher-Reeves yöntemi ile elde edilen çözümlerin, Hardy Cross tekniği sonuçlarıyla karşılaştırılması.

Örnek Şebeke	Kol	Fletcher -Reeves	Hardy Cross	Mutlak Fark	S_q
1	1	4,3851	4,3822	0,0029	0,002
	2	6,8801	6,8818	0,0017	
	3	1,7996	1,8031	0,0035	
	4	6,1847	6,1853	0,0006	
	5	5,0804	5,0787	0,0017	
	6	11,2651	11,2641	0,0010	
	7	11,2651	11,2641	0,0010	
2	1	2,3739	2,3731	0,0008	0,0005
	2	8,2588	8,2591	0,0003	
	3	4,7877	4,7882	0,0005	
	4	7,1617	7,1613	0,0004	
	5	6,3819	6,3822	0,0003	
	6	0,7797	0,7791	0,0006	
	7	3,4712	3,4708	0,0004	
	8	4,2508	4,2501	0,0007	
	9	10,6328	10,6322	0,0006	
	10	10,6328	10,6322	0,0006	
3	1	9,8106	9,8101	0,0005	0,0016
	2	4,3505	4,3491	0,0014	
	3	4,6351	4,6314	0,0037	
	4	4,2914	4,2896	0,0018	
	5	5,4615	5,4611	0,0004	
	6	0,2847	0,2824	0,0023	
	7	-0,3438	-0,3418	0,0020	
	8	5,5206	5,5204	0,0002	
	9	3,3237	3,3223	0,0014	
	10	3,0390	3,0399	0,0009	
	11	2,1377	2,1387	0,0010	
	12	2,4815	2,4805	0,0010	
	13	9,8106	9,8101	0,0005	

6. Adım : Tüm kollar için Q_j değerleri hesaplanır (j=1,2,..,N).

7. Adım : $|Q_i^{k+1} - Q_i^k| < \varepsilon$ ise 2. adıma dönülür,
Tersine durumda işlem bitirilir.

6.4.2. Hessien matrisinin oluşturulması

Hessien matrisi, amaç fonksiyonunun değişkenlere göre ikinci kısmi türevlerinin alınmasıyla oluşturulur. Bu işlem küçük şebekeler için elle yapılabilmekle birlikte, büyük şebekeler için elle çözüm olanaksız olmaktadır.

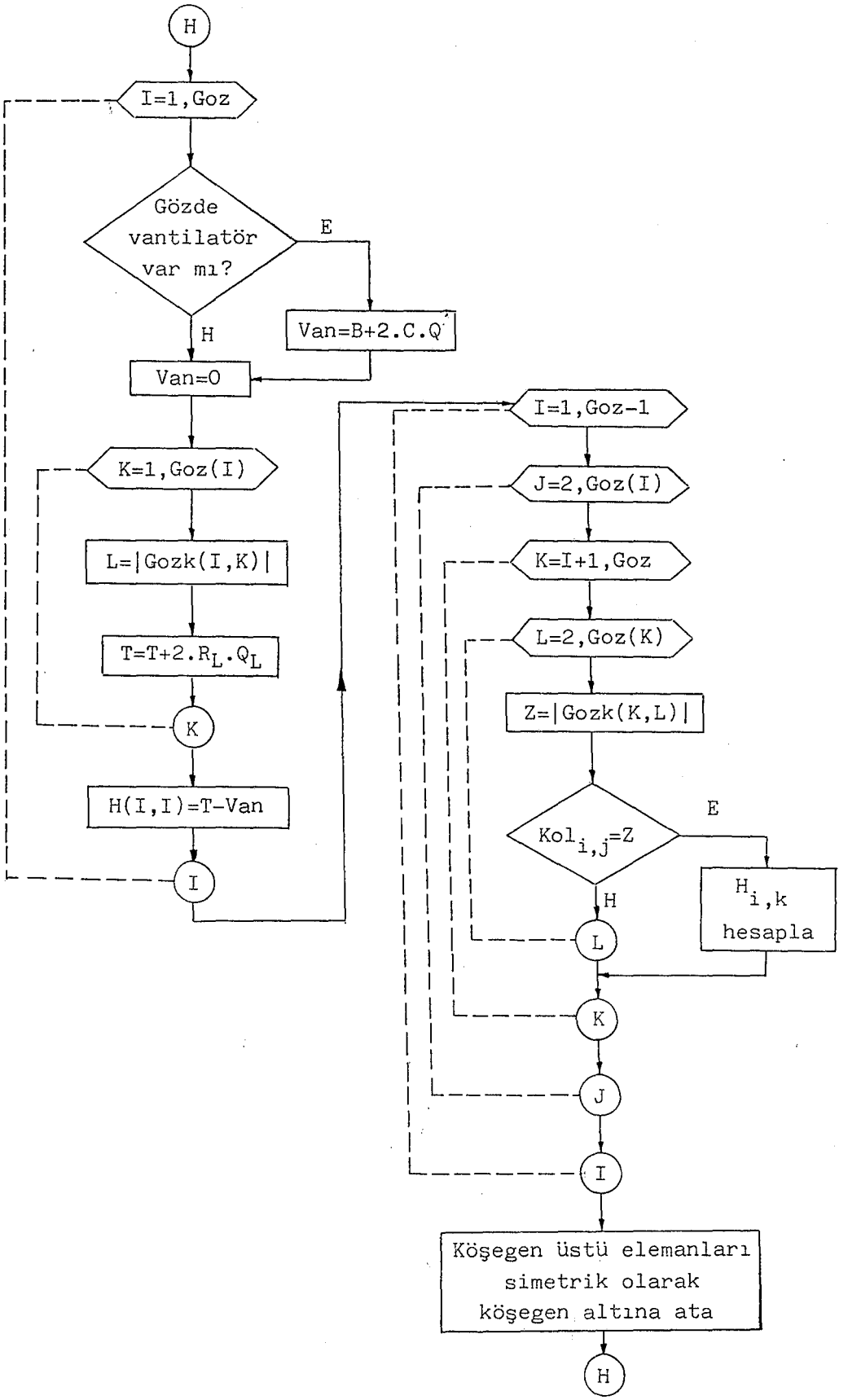
Amaç fonksiyonunun ikinci kısmi türevleri, göz dizileri ile belli bir ilişki içinde bulunduğundan, bilgisayarda göz dizileri taratılarak Hessien matrisi sayısal olarak oluşturulmuş ve (GozXGoz) boyutlu bir H matrisi ile hafızada depolanmıştır. Bu uygulama ile büyük şebekelerin de Newton yöntemi ile çözümlenebilmesi sağlanmıştır.

Newton yöntemi için yazılan bilgisayar programının içinde yer verilen ve Hessien matrisini oluşturan program kesiminin genelleştirilmiş akış şeması Şekil 6,7'de verilmektedir.

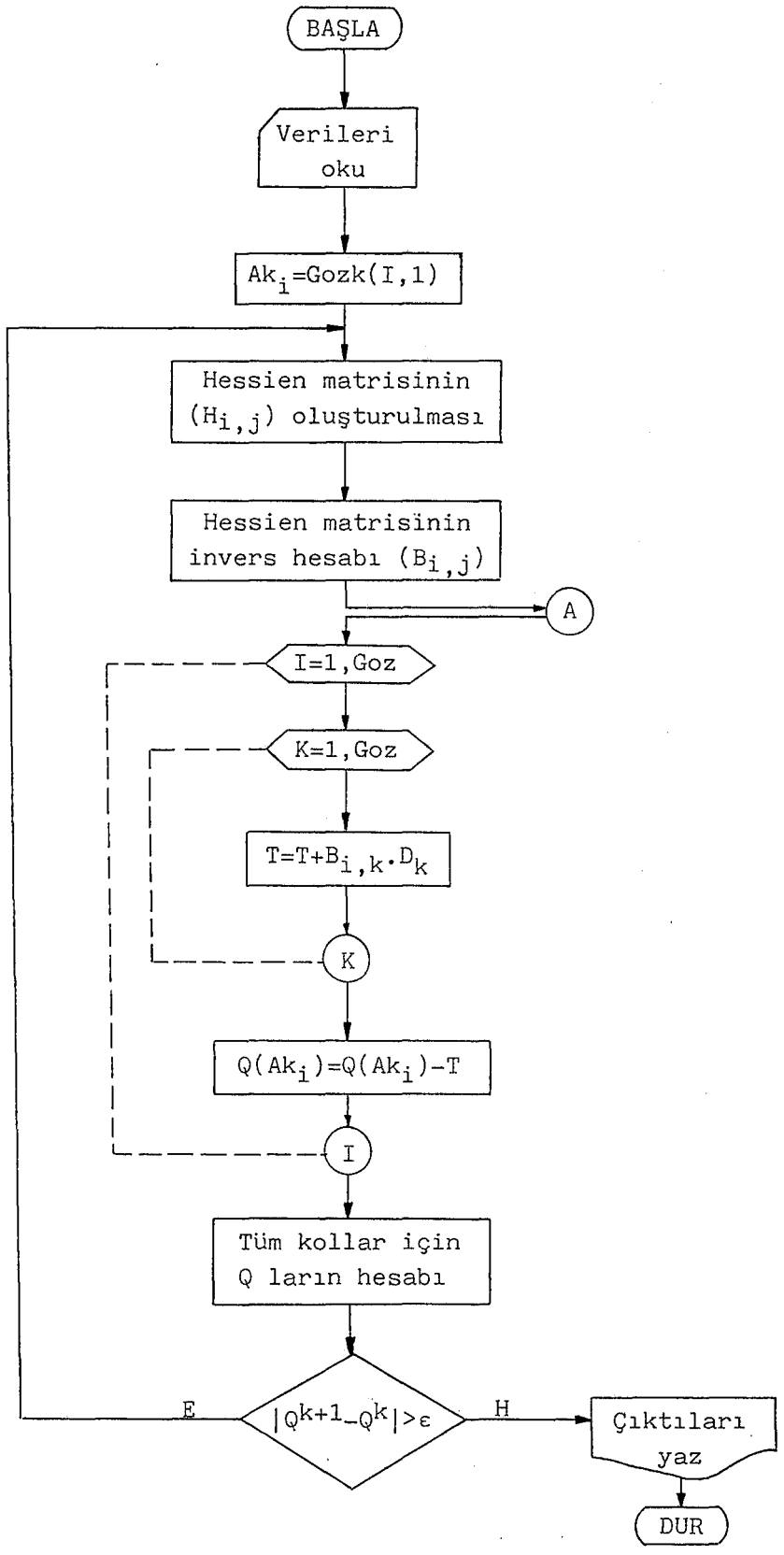
Amaç fonksiyonunun bulunulan noktadaki Hessien matrisi oluşturulduktan sonra, bu matrisin inversinin hesaplanması gerekir. İnvers alma işlemi, Newton yönteminin en güç ve zaman alıcı bölümüdür. Temel matematik kitaplarında verilen invers hesaplama yöntemleri arasında, en yaygın olarak kullanılanı ve bilgisayar programlamaya en yatkın olanı "Eleme Tekniği"dir. Bu teknik için Topçu (1987) tarafından hazırlanan bilgisayar programı, probleme uyarlanarak aynen kullanılmıştır.

6.4.3. Bilgisayar programı

Newton yöntemi için, verilen algoritmayı izleyen bir bilgisayar programı yazılarak örnek şebekeler için denenmiş ve işlerliği kanıtlanmıştır. Programın genelleştirilmiş akış şeması Şekil 6,8'de, program



Şekil 6.7 : Hessien matrisini oluşturan program kesiminin genelleştirilmiş akış şeması.



Şekil 6.8 : Newton yöntemi bilgisayar programının genelleştirilmiş akış şeması.

ise EK-9'da verilmiştir.

Programda göz dizileri taratılarak bulunulan noktadaki Hessien matrisi sayısal olarak oluşturulduktan sonra, bu matrisin inversi hesaplanmaktadır. Algoritmanın dördüncü adımında ise gradyan hesaplama alt programına gidilerek gradyan vektörü belirlenmekte, ters matris ve gradyan vektörü kullanılarak ardışık nokta, beşinci adımda hesaplanmaktadır.

Hessien matrisinin inversinin var olabilmesi bakımından, başlangıç Q° değerleri için, akımın korunumu ilkesine uymak koşuluyla sıfırdan farklı değerler atamak gerekmektedir. Bu nedenle diğer yöntemlerin bilgisayar programlarından farklı olarak, kollardaki direnç değerlerinden hemen sonra, atanan Q° değerleri veri olarak okutulmalıdır. Örnek Şebeke-1 için veri okutma düzeni EK-9'da verilen programdaki gibi olmaktadır. Çıktı olarak ise, iterasyon sayısı, toplam çözüm süresi ve kollardaki hava dağılımı değerleri alınmaktadır.

6.4.4. Optimum çözüm seti

Newton yöntemi için yazılan bilgisayar programı örnek şebekeler için çalıştırılarak işlerliği kanıtlanmış ve elde edilen sonuçlar test edilmiştir. Örnek Şebeke-1 için yöntemin optimum çözüm setine yaklaşımı Çizelge 6,6'da, optimum noktaların Hardy Cross tekniği sonuçlarıyla karşılaştırılması ise Çizelge 6,7'de verilmiştir.

Alınan sonuçlar, Hessien matrisinin inversinin var olması durumunda, havalandırma şebekelerindeki hava dağılımlarının Newton Yöntemi ile hassas bir şekilde hesaplanabileceğine işaret etmektedir.

Newton yöntemi ile, gerek optimum noktaya sağlanan yaklaşım özellikleri, gerekse optimum çözüm seti olarak elde edilen değerler, Hardy Cross yöntemine büyük benzerlikler göstermiştir. Bu özellik, her iki yöntemin de aynı tip matematiksel teknik oluşundan kaynaklanmaktadır.

Çizelge 6.6 : Örnek Şebeke-1 için Newton yönteminin işleyişi.

İterasyon (k)		0	1	2	3	4	5	6	7
Q^k	1	1	0,5	1,7175	3,4091	4,2878	4,3878	4,3821	4,3822
	2	0	38,348	19,2221	10,1806	7,2435	6,8901	6,8817	6,8817
	3	0	0	0,0347	0,4078	1,3752	1,7868	1,8031	1,8031
	4	1	0,5	1,7523	3,8169	5,6631	6,1675	6,1851	6,1852
	5	0	38,348	19,1873	9,7727	5,8683	5,1032	5,0787	5,0787
	6	1	38,848	20,9396	13,5896	11,5314	11,2708	11,2639	11,2639
	7	1	38,848	20,9396	13,5896	11,5314	11,2708	11,2639	11,2639
$\nabla f(Q^k)$	1	1	-587,91	-144,81	-29,85	-2,75	-0,055	0,0004	
	5	-0,6	1323,4	329,5	77,21	11,61	0,361	0,0003	
	6	-484,1	4818,6	1094,4	194,79	18,08	0,417	0,0001	

Çizelge 6.7 : Newton yöntemi ile elde edilen çözümlerin, Hardy Cross tekniği sonuçlarıyla karşılaştırılması.

Örnek Şebeke	Kol	Newton	Hardy Cross	Mutlak Fark	S_q
1	1	4,3821	4,3822	0,0001	0,0001
	2	6,8817	6,8818	0,0001	
	3	1,8030	1,8031	0,0001	
	4	6,1852	6,1853	0,0001	
	5	5,0787	5,0787	0	
	6	11,2639	11,2641	0,0002	
	7	11,2639	11,2641	0,0002	
2	1	2,3731	2,3731	0	0,0001
	2	8,2590	8,2591	0,0001	
	3	4,7881	4,7882	0,0001	
	4	7,1613	7,1613	0	
	5	6,3821	6,3822	0,0001	
	6	0,7792	0,7791	0,0001	
	7	3,4708	3,4708	0	
	8	4,2501	4,2501	0	
	9	10,6322	10,6322	0	
	10	10,6322	10,6322	0	
3	1	9,8102	9,8101	0,0001	0,004
	2	4,3497	4,3491	0,0006	
	3	4,6355	4,6314	0,0041	
	4	4,2851	4,2896	0,0045	
	5	5,4605	5,4611	0,0006	
	6	0,2858	0,2824	0,0004	
	7	-0,3505	-0,3418	0,0087	
	8	5,5251	5,5204	0,0047	
	9	3,3236	3,3223	0,0013	
	10	3,0378	3,0399	0,0021	
	11	2,1368	2,1387	0,0019	
	12	2,4873	2,4805	0,0068	
	13	9,8102	9,8101	0,0001	

6.5. Davidon-Fletcher-Powell Yöntemi

6.5.1. Algoritma

Davidon-Fletcher-Powell (D-F-P) yönteminin havalandırma şebekelerine uygulanmasında aşağıda verilen algoritma izlenmiştir:

1. Adım : Q° başlangıç noktası ve istenen hassasiyet değeri (ε) seçilir, $k=0$ olarak alınır.

2. Adım : (GXG) boyutlu H birim matrisi oluşturulur.

3. Adım : $\nabla f(Q_i^k)$ hesaplanır (i=1,2,..,G).

4. Adım : $P^k = -H^k . \nabla f(Q_i^k)$ arama vektörü belirlenir.

5. Adım : $r^k = \min f[|(Q_j^k - r^k . P_j^k)|]$, (j=1,2,..,N)
yapan optimal adım boyutu Kareli İnterpolasyon tekniği ile hesaplanır.

6. Adım : Ardışık Q noktası,

$$Q_i^{k+1} = Q_i^k + r^k . P^k$$

ifadesi ile saptanır.

7. Adım : $|Q_i^{k+1} - Q_i^k| < \varepsilon$ ise işlem bitirilir.

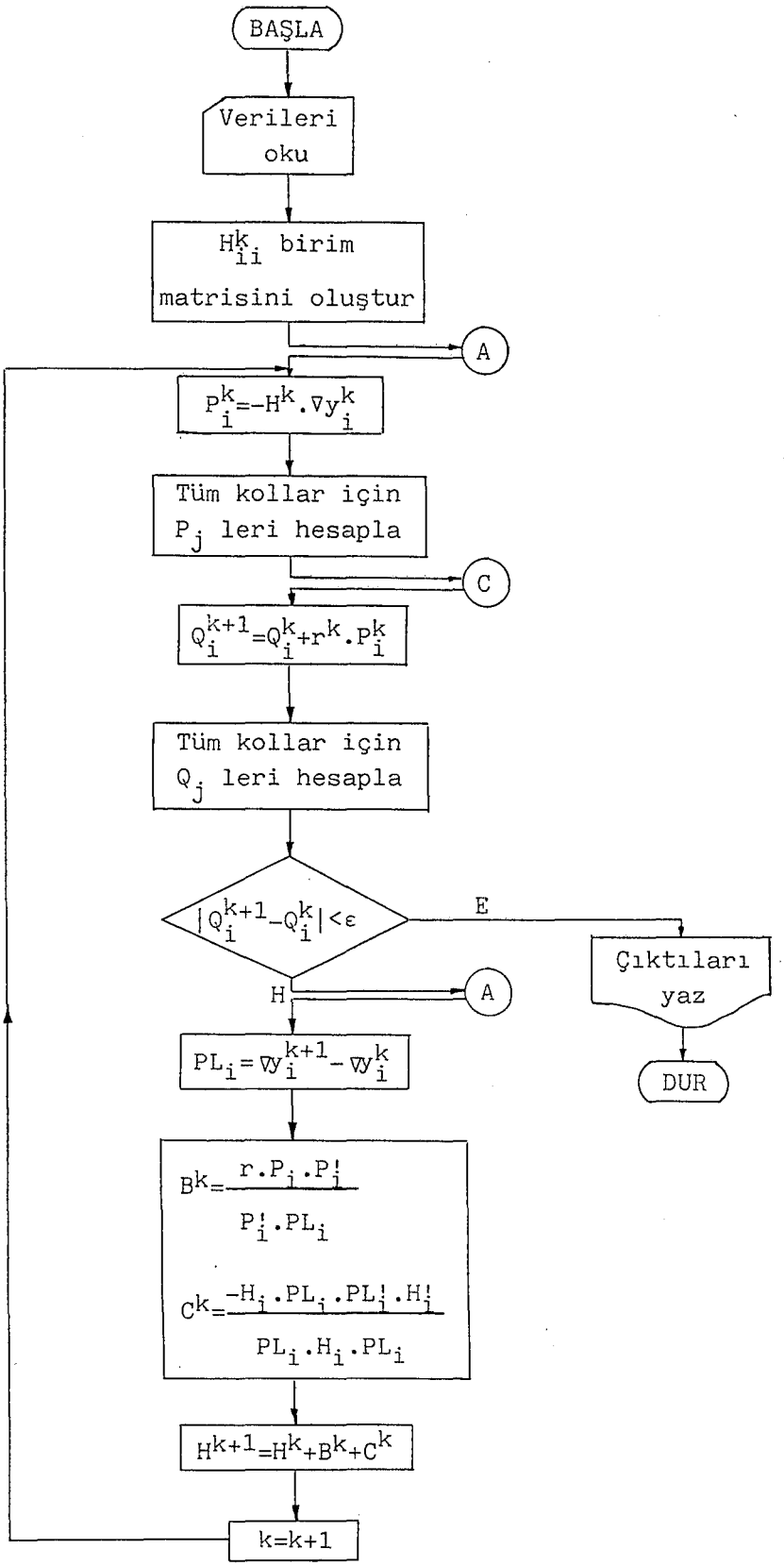
8. Adım : $\nabla f(Q_i^{k+1})$ hesaplanır.

9. Adım : Y^k ve Z^k matrisleri (4-29) ve (4-30) ifadelerinden hesaplandıktan sonra,

$$H^{k+1} = H^k + Y^k + Z^k$$

eşitliğinden yeni H matrisi oluşturulur,

$k=k+1$ yapılıp, 4. adıma dönülür.



Şekil 6.9 : Davidon-Fletcher-Powell yöntemi bilgisayar programının genelleştirilmiş akış şeması.

Programdan çıktı olarak ise, iterasyon sayısı, toplam çalışma süresi ve ele alınan havalandırma şebekesinin kollarındaki hava dağılımı değerleri alınmaktadır.

6.5.3. Optimum çözüm seti

Yöntemin uygulanmasında başlangıç Q^0 değerleri sıfır olarak atanmış, hassasiyet değeri ise 0,005 m^3/s olarak alınmıştır. İterasyon bitirme ölçütü olarak, H matrisi elemanlarının veya ulaşılan noktada hesaplanan gradyan vektörünün kullanılması mümkün olabilmekle birlikte, diğer yöntemlere uygunluk sağlamak için $|Q^{k+1}-Q^k|$ farkı bu amaçla kullanılmıştır.

Örnek Şebeke-1'in D-F-P yöntemi ile çözümlenmesinde, her iterasyonda ulaşılan Q^k noktaları Çizelge 6,8'deki gibi olmuştur. Tüm örnek şebekeler için belirlenen optimum çözüm setlerinin, Hardy Cross tekniği ile elde edilen sonuçlarla karşılaştırılması ise Çizelge 6,9'da verilmiştir.

Verilen sonuçlar, havalandırma şebekeleri için oluşturulan kısıtsız modelin D-F-P yöntemi ile enküçüklenerek, kollardaki hava dağılımının belirlenebileceğini kanıtlamaktadır.

En Hızlı İniş ve Fletcher-Reeves yöntemlerine oranla, Davidon-Fletcher-Powell yöntemi ile elde edilen sonuçlar, Hardy Cross tekniği sonuçlarına daha yakın olmuş, tahminin standard hatası değerleri daha düşük olarak hesaplanmıştır. Ayrıca optimum çözüme ulaşmak için yapılan iterasyon sayısı da, daha az olmuştur. Ancak yöntemde yoğun matris işlemleri yapıldığından bir iterasyon için harcanan süre, dolayısıyla toplam çözüm süresi Fletcher-Reeves yönteminden daha fazla olmuştur. Bu gözlemler, literatür ile büyük bir uyum göstermektedir.

Çizelge 6.8 : Örnek Şebeke-1 için Davidon-Fletcher-Powell yönteminin işleyişi.

İterasyon (k)		0	1	2	3	4	5	6
Q^k	1	0	0	4,005	4,8941	4,4387	4,3834	4,3833
	2	0	9,2716	6,6339	5,8957	6,8411	6,8805	6,8806
	3	0	9,2716	0,9601	0,7673	1,6886	1,8026	1,8027
	4	0	9,2716	4,9651	5,6614	6,1273	6,1861	6,1859
	5	0	0	5,6737	5,1284	5,1525	5,0778	5,0779
	6	0	9,2716	10,6389	10,7898	11,2798	11,2639	11,2639
	7	0	9,2716	10,6389	10,7898	11,2798	11,2639	11,2639
P^k	1	0	48,226	7,9335	-0,9114	-0,0692	-0,0013	
	2	494,54	-31,76	-6,5871	1,8923	0,0493	0,0012	
	3	494,54	-100,08	-1,7209	1,8441	0,1426	0,0003	
	4	494,54	-51,86	6,2126	0,9326	0,0734	-0,0009	
	5	0	68,32	-4,8661	0,0482	-0,0933	0,0008	
	6	494,54	16,46	1,3464	0,9809	-0,0199	-0,0001	
	7	494,54	16,46	1,3464	0,9809	-0,0199	-0,0001	
r^k		0,0187	0,0881	0,112	0,4996	0,7997	0,125	

Çizelge 6.9 : Davidon-Fletcher-Powell yöntemi ile elde edilen çözümlerin, Hardy Cross tekniği sonuçlarıyla karşılaştırılması.

Örnek Şebeke	Kol	D-F-P	Hardy Cross	Mutlak Fark	S_q
1	1	4,3833	4,3822	0,0011	0,0007
	2	6,8806	6,8818	0,0012	
	3	1,8027	1,8031	0,0004	
	4	6,1859	6,1853	0,0006	
	5	5,0779	5,0787	0,0008	
	6	11,2639	11,2641	0,0002	
	7	11,2639	11,2641	0,0002	
2	1	2,3637	2,3731	0,0094	0,0083
	2	8,2629	8,2591	0,0038	
	3	4,7861	4,7882	0,0021	
	4	7,1497	7,1613	0,0116	
	5	6,3665	6,3822	0,0156	
	6	0,7831	0,7791	0,0040	
	7	3,4769	3,4708	0,0061	
	8	4,2601	4,2501	0,0100	
	9	10,6267	10,6322	0,0055	
	10	10,6267	10,6322	0,0055	
3	1	9,8117	9,8101	0,0016	0,0032
	2	4,3462	4,3491	0,0029	
	3	4,6323	4,6314	0,0009	
	4	4,2867	4,2896	0,0029	
	5	5,4654	5,4611	0,0043	
	6	0,2860	0,2824	0,0036	
	7	-0,3455	-0,3418	0,0037	
	8	5,5249	5,5204	0,0045	
	9	3,3281	3,3223	0,0058	
	10	3,0420	3,0399	0,0021	
	11	2,1374	2,1387	0,0013	
	12	2,4829	2,4805	0,0024	
	13	9,8117	9,8101	0,0016	

7. ÇÖZÜM TEKNİKLERİNİN KARŞILAŞTIRILMASI

Havalandırma şebekelerinin analizinde herhangi bir yöntemle çözüme ulaşmak tek başına yeterli olmayıp, uygulamacı maden mühendisine, geleneksel Hardy Cross yöntemine oranla hesaplama kolaylıkları getirebilmek önem taşımaktadır. Bu nedenle, havalandırma şebekelerinin kısıtsız optimizasyon teknikleriyle çözümlenebileceği kanıtlandıktan sonra, ele alınan bu tekniklerin çalışma performansları karşılaştırılmış, Hardy Cross tekniğine üstünlük sağlayıp sağlayamadıkları test edilmiştir.

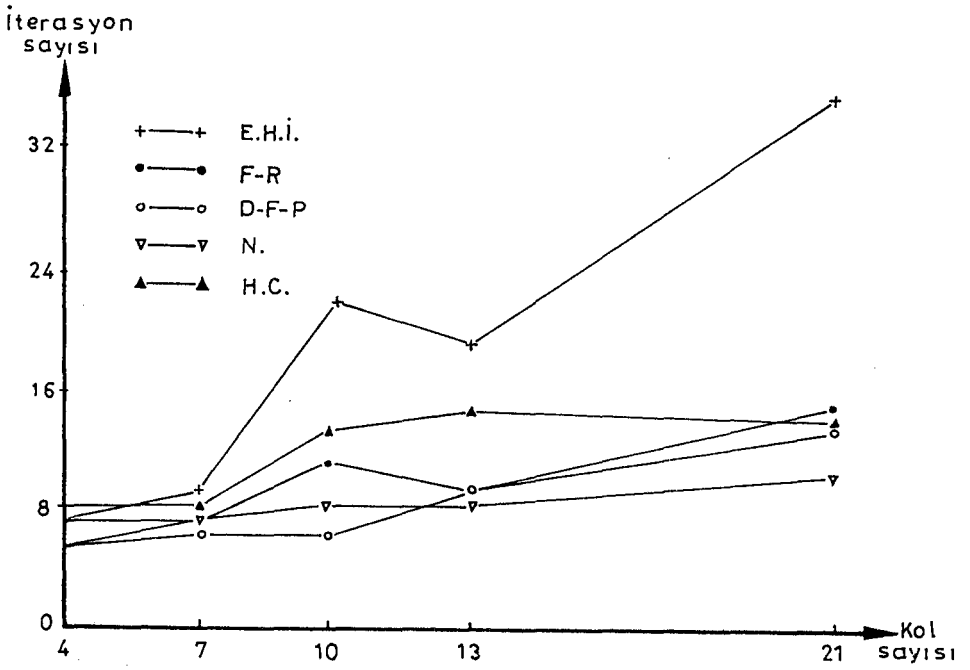
Karşılaştırma ölçütleri olarak, toplam iterasyon sayısı, çözüm süresi, kullanılan bilgisayar belleği ve optimum çözüme yaklaşım özellikleri kullanılmıştır. Örnek şebekeler, ele alınan teknikler için yazılan bilgisayar programları yardımıyla çözümlenmiş, karşılaştırma ölçütleri bilgisayar çıktısı olarak alınmıştır. Alınan sonuçlar sayısal ve grafiksel olarak değerlendirilmiştir. Uygulanan tekniklerin örnek şebekeler için çözüme ulaşma özellikleri Çizelge 7,1'de topluca verilmiştir.

Çizelge 7.1 : Örnek şebekeler için çözüme ulaşma özellikleri.

	Şebeke	E.H.İ.	F-R	D-F-P	Newton	H.C.
İterasyon sayısı	1	8	7	6	7	8
	2	22	11	6	8	13
	3	19	9	9	8	15
	4	35	15	13	9	14
Çözüm süresi (s)	1	21	19	22	16	9
	2	73	39	30	32	21
	3	93	45	71	57	28
	4	290	148	181	168	52
Kullanılan bellek (B)	1	4009	4863	5249	3943	1766
	2	4028	4902	5259	3974	1798
	3	4044	4946	5313	4004	1843
	4	4209	5163	5668	4203	2123

7.1. İterasyon Sayısı

Kısıtsız optimizasyon tekniklerinin, göz önüne alınan örnek şebekelere uygulanmasında, çözüme ulaşmak için gereken iterasyon sayılarının şebekedeki kol sayısına göre değişimi Şekil 7,1'de verilmiştir.



Şekil 7.1 : Şebekedeki kol sayısı ile iterasyon sayısı arasındaki ilişki.

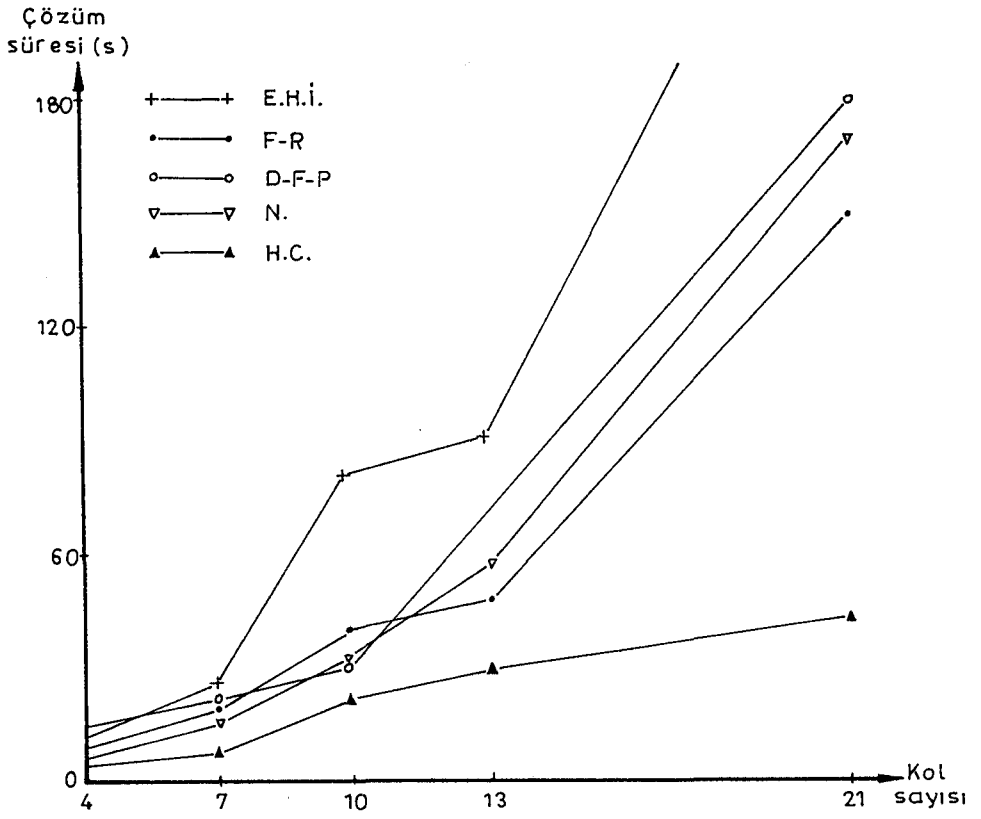
En Hızlı İniş yönteminin küçük şebekeler için bile, optimum çözüme fazla sayıda iterasyonda ulaşabildiği gözlemlenmektedir. Bu durum, optimum nokta civarında bu yöntemin zigzag şeklinde bir ilerleme yapmasından kaynaklanmaktadır. Diğer optimizasyon teknikleri, küçük şebekeler için Hardy Cross tekniğinden daha az iterasyonda sonuca gitmekle beraber, kol sayısı arttıkça bu üstünlük kaybolmaktadır.

7.2. Çözüm Süresi

Her bir iterasyonda yapılan işlemlerin her yöntemde çok farklı olması ve değişik sürelerde tamamlanması, yöntemlerin karşılaştırılmasında

iterasyon sayısının tek başına etkili bir parametre olarak alınmasını engellemektedir. Bu nedenle, çözüm için harcanan sürelerin karşılaştırılması daha anlamlı olmaktadır.

Kol sayısına bağlı olarak, uygulanan çözüm teknikleriyle optimum noktaya ulaşmada harcanan sürelerin değişimi Şekil 7,2'de grafiksel olarak verilmiştir.



Şekil 7.2 : Kol sayısına bağlı olarak çözüm süresinin değişimi.

Çözüm süresi bakımından hiç bir optimizasyon tekniğinin Hardy Cross yöntemine üstünlük sağlayamadığı görülmektedir. En Hızlı İniş yönteminde iterasyon sayısının fazla olmasıyla uyumlu olarak çözüm süresi de oldukça yüksek olmaktadır. Küçük şebekelerde bile çözüm süresi fazla olan bu yöntemin havalandırma şebekelerinin analizinde alternatif bir yöntem

olmaktan çok uzak olduğu anlaşılmaktadır.

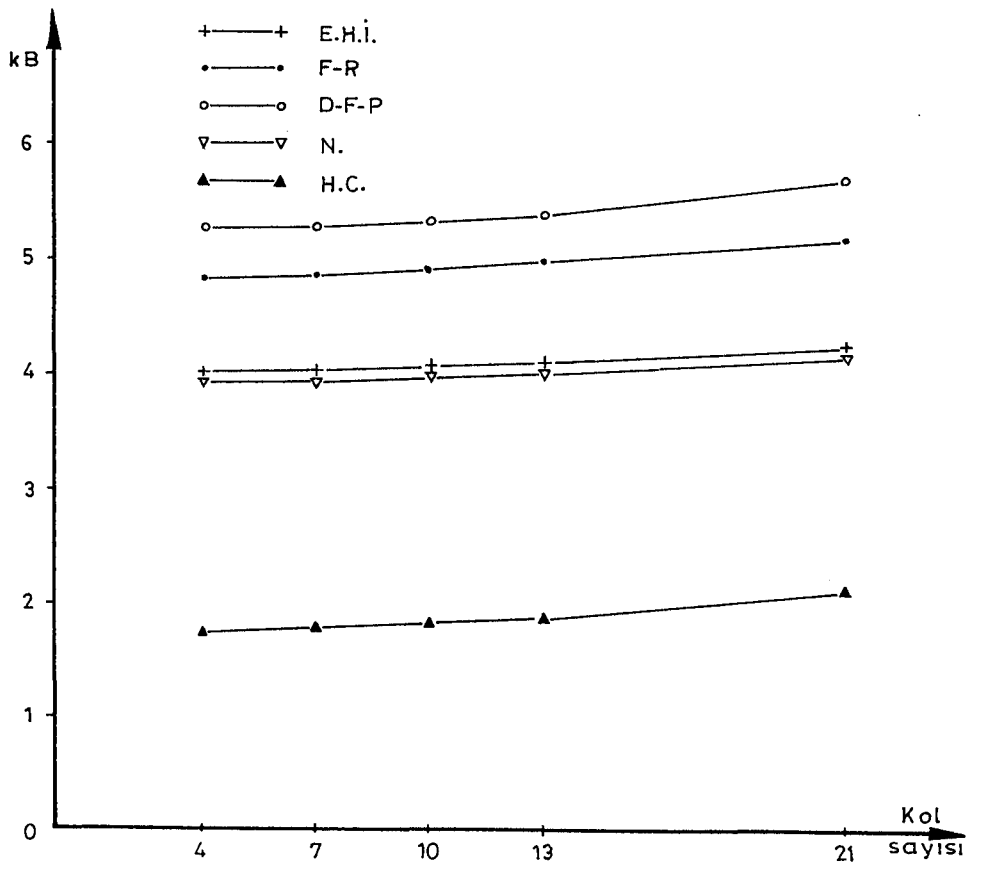
Diğer kısıtsız optimizasyon teknikleri arasında, Fletcher-Reeves yöntemi şebekedeki kol sayısı arttıkça avantajlı duruma geçmekle birlikte, çözüm süresi bakımından Hardy Cross yöntemine rakip olamamaktadır. İterasyon sayısının az olmasına karşın çözüm sürelerinin fazla olması, D-F-P yönteminde matris işlemlerinin yoğun olmasından, Newton yönteminde ise Hessien matrisinin oluşturulmasının ve bu matrisin inversinin alınmasının fazla zaman almasından kaynaklanmaktadır.

Uygulanan kısıtsız optimizasyon tekniklerinin çözüme ulaşmak için harcadıkları sürelerin, Hardy Cross yöntemine oranla çok daha fazla olmasının en büyük nedeni, her iterasyonda ilerleme yönünde tek boyutlu arama yapma zorunluluğunun olmasıdır. Her bir iterasyonun tamamlanması için harcanan sürenin çok büyük kısmı, doğrusal arama alt programında optimal adım boyutunun belirlenmesi sırasında harcanmaktadır. Optimal adım boyutunun belirlenmesinde en hızlı sonucu veren Kareli İnterpolasyon yönteminde bile, birden fazla iterasyonda bu değer belirlenebilmektedir. Bir başka deyişle kısıtsız optimizasyon tekniklerinin bir iterasyonunda, aslında birden fazla tekrarlı işlem yapılmakta, dolayısıyla çözüm süresi yüksek olmaktadır.

7.3. Kullanılan Bellek Hacmı

Bilgisayar belleğinde kullanılan hacım, şebekedeki kol sayısına ve uygulanan çözüm yöntemine göre değişmektedir. Örnek şebekelerin çözümlenmesinde gözlemlenen değişim Şekil 7,3'de verilmiştir.

Alınan sonuçlara göre, kullanılan bellek hacmı bakımından da kısıtsız optimizasyon teknikleri Hardy Cross yöntemine üstünlük sağlayamamaktadır. Bu tekniklerde her iterasyonda gradyan vektörünün hesaplanması, doğrusal arama yapılması, matris işlemlerinin yoğun olması ve bu değerlerin hafızada depolanma zorunluluğu bu sonucu yaratmaktadır.

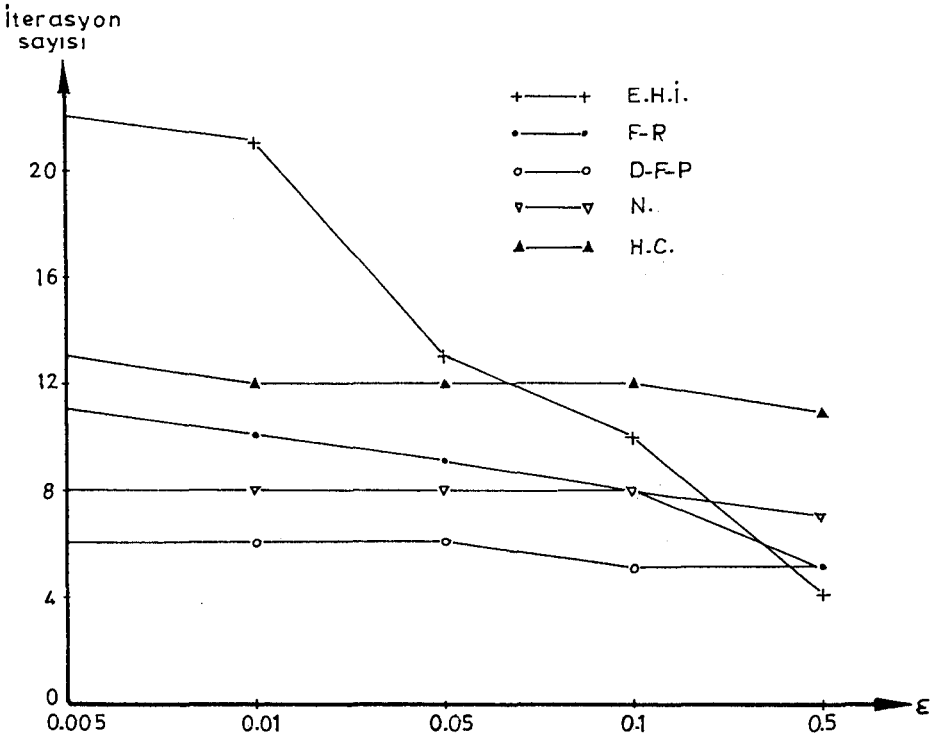


Şekil 7.3 : Şebekedeki kol sayısına ve kullanılan çözüm tekniğine göre, kullanılan bellek hacmı.

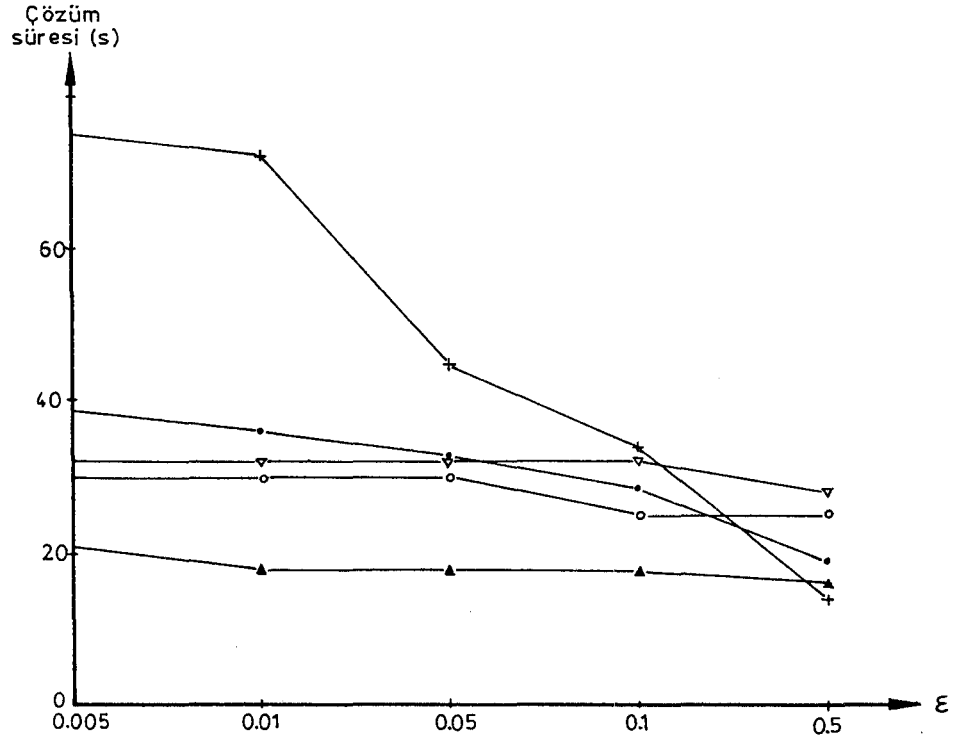
7.4. Hassasiyet Değerinin Çalışma Performansına Etkisi

Kısıtsız optimizasyon tekniklerinin örnek şebekelere uygulanmasında, hassasiyet değeri $\varepsilon=0,005\ m^3/s$ olarak alınmış, karşılaştırmalarda bu hassasiyet derecesi ile alınan sonuçlar kullanılmıştır. Daha düşük hassasiyet derecesinde çalışmanın yöntemlerin çalışma performanslarına etkisini irdelemek amacıyla, örnek şebekeler çeşitli hassasiyet değerleri için çözümlenmiş, alınan sonuçlar Şekil 7,4-7,9'da grafiksel olarak yorumlanmıştır.

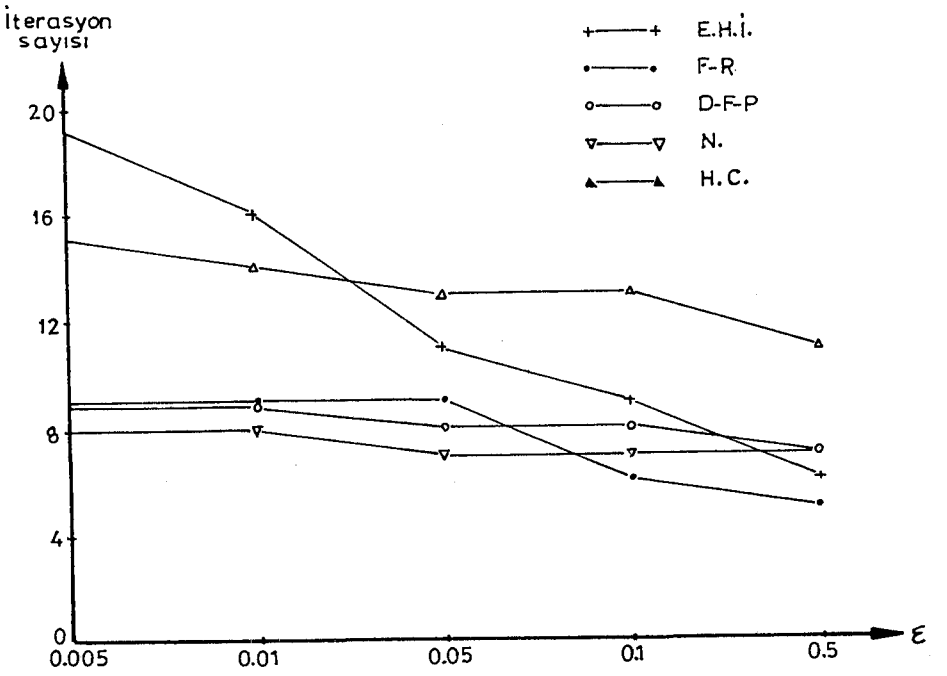
Şekillerin incelenmesinden, hassasiyet değerinin daha büyük seçilmesi, bir başka deyimle hesaplama hassasiyetinin azaltılması ile, Hardy Cross ve Newton yöntemlerinin çalışma performanslarında fazla bir değiş-



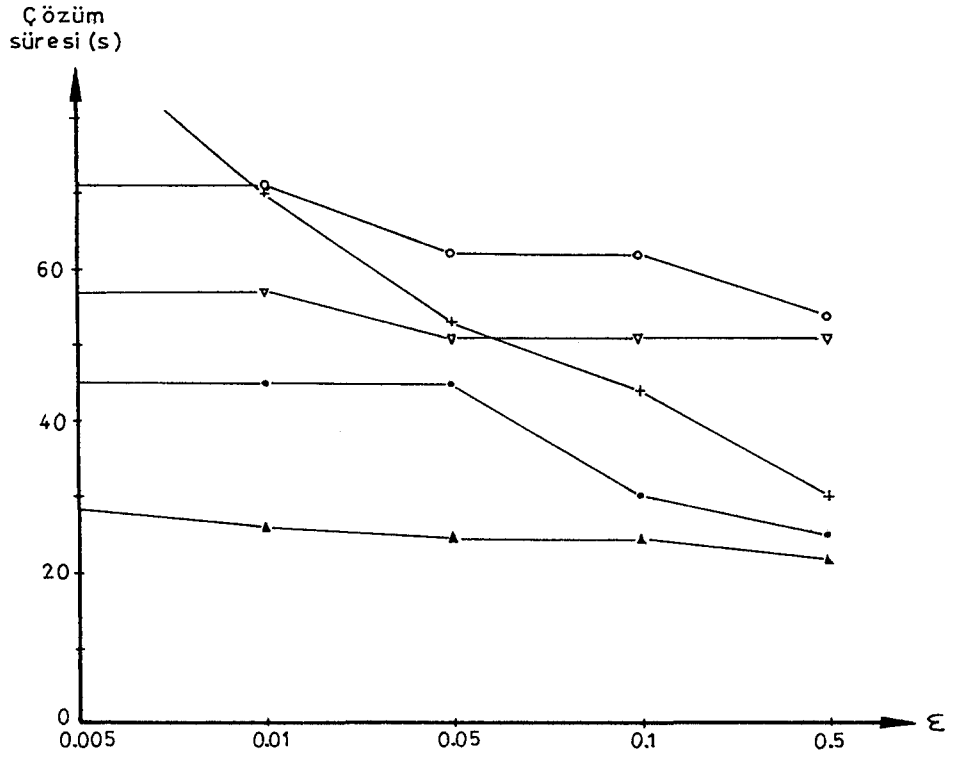
Şekil 7.4 : Örnek Şebeke-2 için hassasiyet değeri ile iterasyon sayısının değişimi



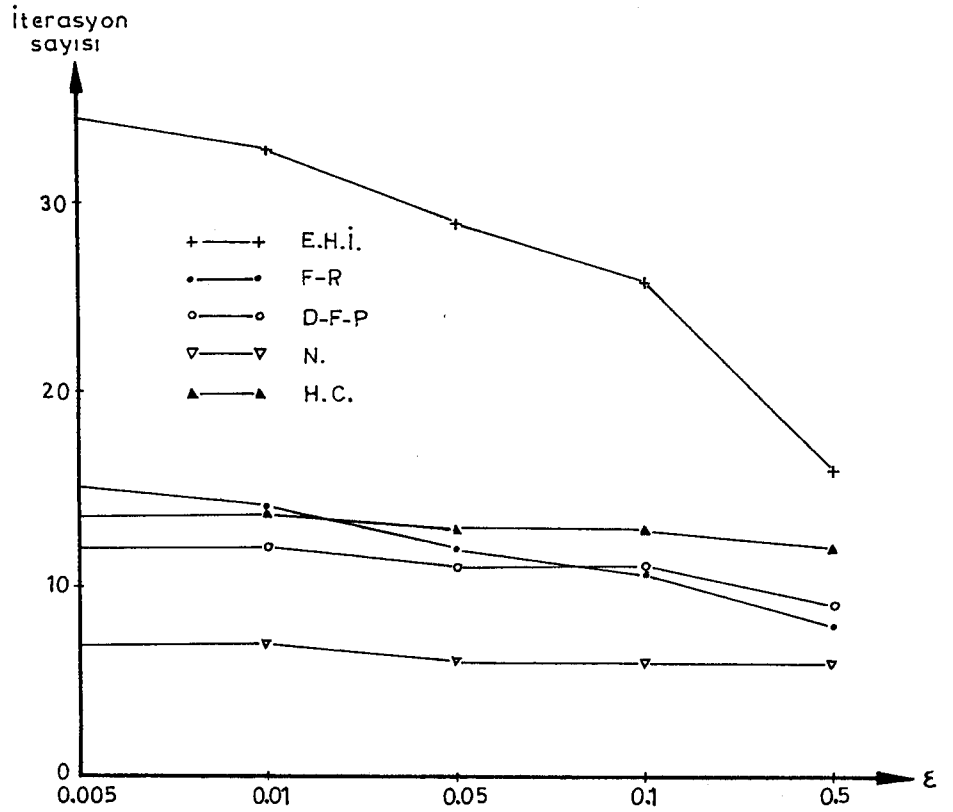
Şekil 7.5 : Örnek Şebeke-2 için hassasiyet değeri ile çözüm süresinin değişimi



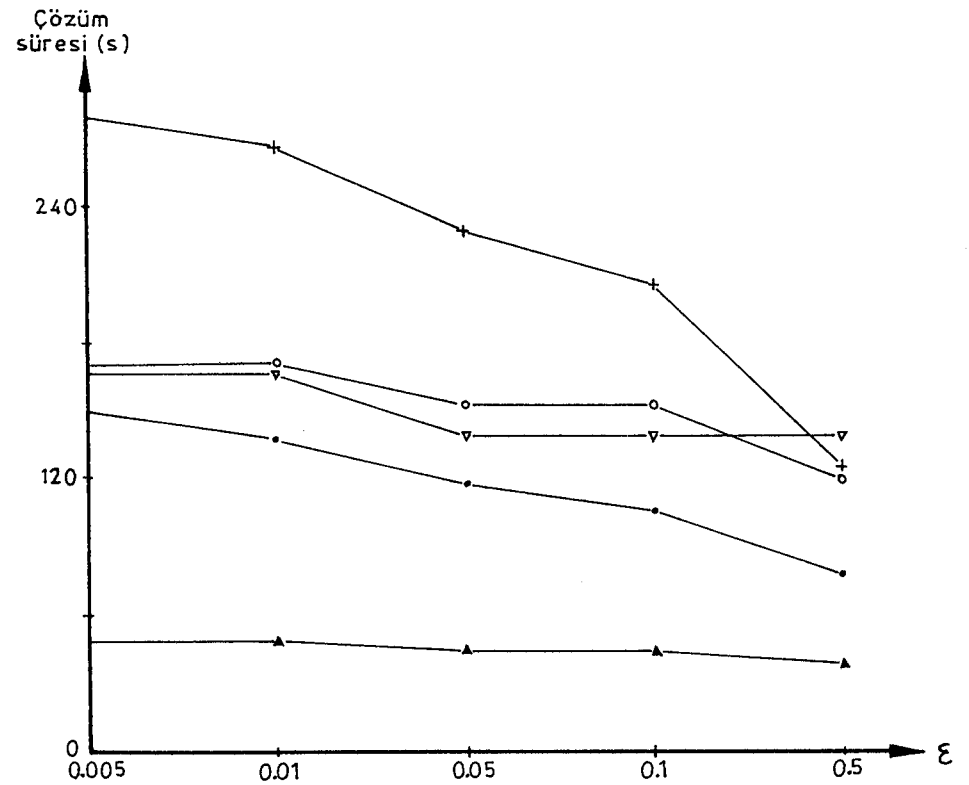
Şekil 7.6 : Örnek Şebeke-3 için hassasiyet değeri ile iterasyon sayısının değişimi.



Şekil 7.7 : Örnek Şebeke-3 için hassasiyet değeri ile çözüm süresinin değişimi.



Şekil 7.8 : Örnek Şebeke-4 için hassasiyet değeri ile iterasyon sayısının değişimi.



Şekil 7.9 : Örnek Şebeke-4 için hassasiyet değeri ile çözüm süresinin değişimi.

şiklik olmadığı, diğer yöntemlerde ise gerek iterasyon sayısı, gerekse çözüme ulaşma süresi bakımından hızlı bir iyileşme sağlandığı anlaşılmaktadır. Bu durum, optimum nokta yakınlarında klasik tekniklerin hızlı bir yakınsama özelliğine sahip olması, kısıtsız optimizasyon tekniklerinin ise optimum noktaya çok çabuk yaklaşmalarına karşın, bu bölgede yavaş bir yakınsama göstermeleri ile açıklanabilir.

Hesaplama hassasiyeti azaldıkça Fletcher-Reeves yöntemi Hardy Cross tekniğine rakip olarak belirmekle birlikte, yine de açık bir üstünlüğe ulaşamamaktadır.

7.5. Göz Dizileri Seçiminin Çalışma Performansına Etkisi

Hardy Cross tekniğinde göz dizilerinin seçiminin iterasyon sayısı ve çözüm süresi üzerindeki etkisi büyüktür. Büyük dirençleri kolları birden fazla göz içinde kullanmak, iterasyon sayısını, dolayısıyla çözüm süresini artırmaktadır. Bu olumsuzluğu önlemek için göz dizilerinin seçiminde 2.4.2. bölümünde verilen kurallara uyulmalıdır.

Kısıtsız optimizasyon tekniklerinin havalandırma şebekelerine uygulanmasında da, gerek gradyan vektörünün hesaplanmasında, gerekse bağımlı değişkenlerin bağımsız değişkenler cinsinden ifade edilmesinde göz tekniğinden yararlanılmıştır. Göz dizilerinin oluşturulmasında, Hardy Cross tekniği için verilen kurallara aynen uyulmuştur.

Göz dizilerinin seçiminde Hardy Cross tekniği için izlenmesi zorunlu olan kurallara, geliştirilen çözüm tekniklerinde de uyup uymamanın çalışma performansları üzerindeki etkisini araştırmak amacıyla Örnek Şebeke-4'den, değişik göz dizileri oluşturulmuştur (Çizelge 7,2).

Çizelgede verilen göz dizilerinin seçiminde yüksek dirençli kolların ana kol olarak alınması kuralı gözönüne alınmamıştır. Sadece, vantilatör bulunan kollar ana kol olarak alınmış, diğer ana kollar ise rastgele alınmıştır.

Çizelge 7.2 : Örnek Şebeke-4'den seçilen göz dizileri.*

Göz dizisi kodu	Göz no	Göz dizileri
A	1	21 1 3 4 8 9 16 10 14 20
	2	11 -10 -16
	3	15 -14 -10 -16 -9 -8
	4	6 7 -16 -9 -8
	5	13 -9 -8 -4 5
	6	17 19 -20 -14 -10 -16 -9 12
	7	18 19 -20 -14 -10 -16 -9 -8 -4 5
	8	2 7 -16 -9 -8 -4 -3
B	1	21 1 3 5 18 19
	2	2 -6 -4 -3
	3	7 -16 -9 -8 6
	4	10 -11 16
	5	20 -19 -18 -5 4 8 9 11 14
	6	13 -9 -8 -4 5
	7	15 -14 -11 -9 -8 -4 5 18 17
	8	12 17 -18 -5 4 8
C	1	21 1 2 7 10 14 20
	2	17 19 -20 -14 -10 -16 -9 12
	3	18 19 -20 -14 -10 -16 -13
	4	11 -10 -16
	5	3 4 6 -2
	6	8 9 16 -7 -6
	7	5 13 16 -7 -6 -4
	8	15 -14 -10 -16 -9 12
D	1	21 1 2 -6 8 9 11 14 20
	2	7 -16 -9 -8 6
	3	10 -11 16
	4	12 15 -14 -11 -9
	5	3 4 6 -2
	6	17 19 -20 -15
	7	13 11 14 20 -19 -18
	8	5 18 19 -20 -14 -11 -9 -8 -4

* Göz dizilerindeki ilk kollar ana kollardır.

Göz dizilerinin seçiminin yöntemlerin çalışma performansları üzerindeki etkisini incelemek amacıyla Örnek Şebeke-4, seçilen değişik göz dizileri için, ele alınan çözüm teknikleri kullanılarak çözümlenmiş ve elde edilen sonuçlar test edilmiştir (Çizelge 7,3).

Çizelge 7.3 : Örnek Şebeke-4'den seçilen değişik göz dizileri için, yöntemlerin çözüme ulaşma özellikleri.

Göz dizisi kodu		A	B	C	D
Gözlerdeki toplam kol sayısı		54	47	46	45
ΣR_a		14972	10624	4490	1362
ΣR_t		237	4585	10718	13846
$\Sigma R_a / \Sigma R_t$		63,2	2,32	0,42	0,098
H.C.	İterasyon sayısı	14	85	84	87
	Çözüm süresi (s)	52	221	215	220
E.H.İ.	İterasyon sayısı	35	53	57	64
	Çözüm süresi (s)	290	446	468	512
F-R	İterasyon sayısı	15	23	22	25
	Çözüm süresi (s)	148	219	198	209
D-F-P	İterasyon sayısı	12	12	12	12
	Çözüm süresi (s)	169	162	163	164
Newton	İterasyon sayısı	7	7	7	7
	Çözüm süresi (s)	168	151	151	147

ΣR_a : Ana kolların dirençlerinin toplamı,
ΣR_t : Tali kolların dirençlerinin toplamı.

Çizelge değerlerinin incelenmesinden, Hardy Cross yöntemine benzer şekilde En Hızlı İniş ve Fletcher-Reeves yöntemlerinde de, büyük dirençli kolların ana kol olarak alınmaması durumunda, iterasyon sayısı ve çözüm süresinin büyük oranda arttığı gözlenmektedir. Bu yöntemlerin uygulanmasında, göz seçim tekniğine uymak zorunlu olmaktadır.

Çizelge 7,3'ün gösterdiği en ilginç sonuç, Newton ve D-F-P yöntemlerinde göz dizileri düzeninin iterasyon sayısı ve çözüm süresi üzerinde olumsuz bir etkisinin belirmemesidir. Bu durum, Newton yönteminin ikinci kısmi türevleri kullanmasından, D-F-P yönteminde ise Newton yöntemine benzer şekilde Hessien matrisinin inversine yaklaşılmasından kaynaklanmaktadır.

Hardy Cross yönteminde göz dizilerinin seçilmesi özellikle büyük şebekeler için, uzun ve sıkıcı işlemleri gerektirmekte, hata yapma olasılığı sözkonusu olabilmektedir. Bu işlemin bilgisayar yardımıyla yapılması durumunda da, programın çözüme ulaşma süresi yüksek olmaktadır. Yeraltı hava yollarında oluşacak değişiklikler sonucu kol dirençlerinin değişmesi durumunda, göz dizilerinin yeniden seçilmesi gerekecektir.

Newton ve Davidon-Fletcher-Powell yöntemlerinde iterasyon işlemleri için harcanan süre Hardy Cross yönteminden fazla olmakla birlikte, göz seçiminin elle yapılmasında büyük kolaylıklar sağlamaktadır. Bu seçimin bilgisayara yaptırılması durumunda da, toplam çözüm süresi bakımından avantajlı olabilmektedir.

Newton ve D-F-P yöntemleri için önerilen göz seçim işleminde büyük dirençli kolların ana kol olarak alınması ve tali kolların kapalı bir göz oluşturmaması koşullarını sağlamak gerekmemekle birlikte, modelin uygulanabilir olması bakımından aşağıda verilen kurallara uyulmalıdır.

a) G=N-K+1 sayıda göz oluşturulmalıdır.
b) Vantilatör bulunan kollar ana kol olarak alınmalıdır.
c) Tüm kollar en az bir gözde yer almalıdır.
d) Ana kollar sadece bir gözde yer almalı ve bu gözün ilk kolu olarak alınmalıdır.

Newton yönteminin sakıncalarını gözönüne alarak, verilen koşulları gerçekleyecek şekilde oluşturulan göz dizilerini kullanan D-F-P yöntemi, havalandırma şebekelerinin analizine büyük kolaylıklar getirmektedir.

7.6. Başlangıç Q Değerlerinin Etkisi

İteratif yöntemlerde çözüm değerleri aranan değişkenler için, başlangıçta tahmini değerler atanmakta, düzeltme işlemleri bu atanmış değerler üzerinde yapılmaktadır.

Hardy Cross tekniğinde tüm kollar için başlangıç Q değerlerinin akımın korunumu ilkesine uymak koşuluyla, sıfırdan farklı değerler olarak atanması iterasyon sayısını ve çözüm süresini azaltmaktadır(McPherson, 1966). Aynı şekilde, şebekenin kollarındaki hava akım yönlerinin gerçeğe uygun bir şekilde atanması da çözüm süresini iyileştirmektedir.

Kısıtsız optimizasyon tekniklerinin havalandırma şebekelerine uygulanmasında da, ana kollar veya tüm kollar için başlangıç Q değerleri atanmakta ve ilk iterasyonda bu değerler üzerinde düzeltme işlemi yapılmaktadır. Atanan Q^{o} değerlerinin yöntemlerin çalışma performanslarına etkisini araştırmak amacıyla, Örnek Şebeke-4 değişik başlangıç Q değerleri ile çözümlenmiş, alınan sonuçlar Çizelge 7,4'de, grafiksel yorumları ise Şekil 7,10 ve Şekil 7,11'de verilmiştir.

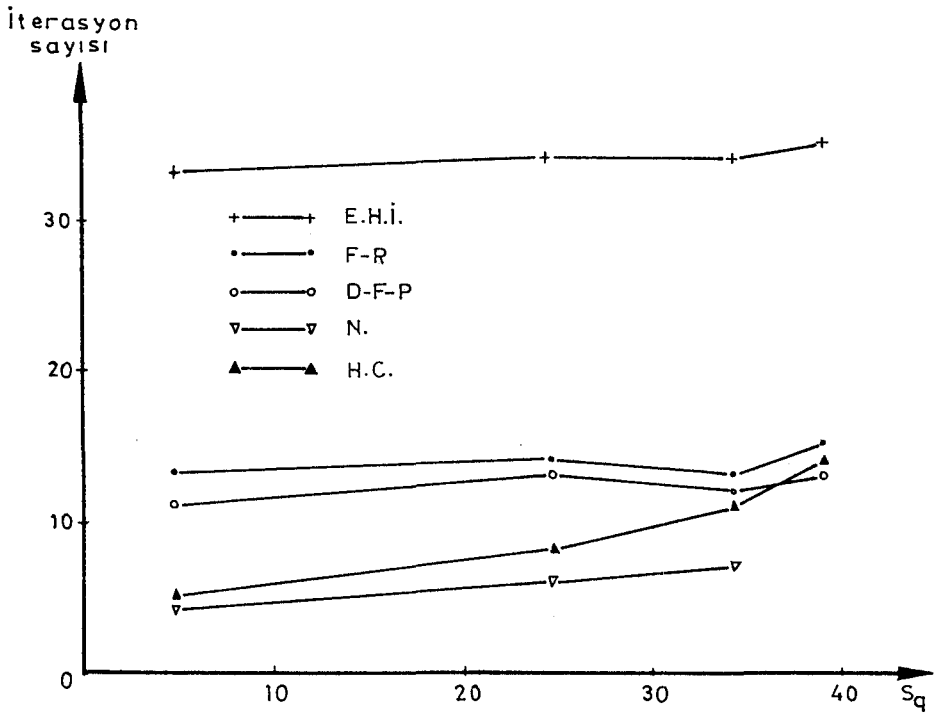
Çizelge ve şekiller, başlangıç Q değerlerinin sıfırdan farklı olarak atanmasının Hardy Cross ve Newton yöntemlerinde iterasyon sayısı ve çözüm süresini büyük oranda azalttığını bir kez daha kanıtlamaktadır.

Aynı uygulama diğer üç yöntemde daha düşük oranda iyileştirmeler sağlamakta, atanan değerlerin optimum çözüme yakınlık derecesi ise belirgin bir etki yaratmamaktadır. Bu sonucun nedeni, bu yöntemlerde ilk bir kaç iterasyonda optimum çözüme çok yaklaşılması, daha sonraki iterasyonlarda ise yakınsama hızının düşük olmasıdır.

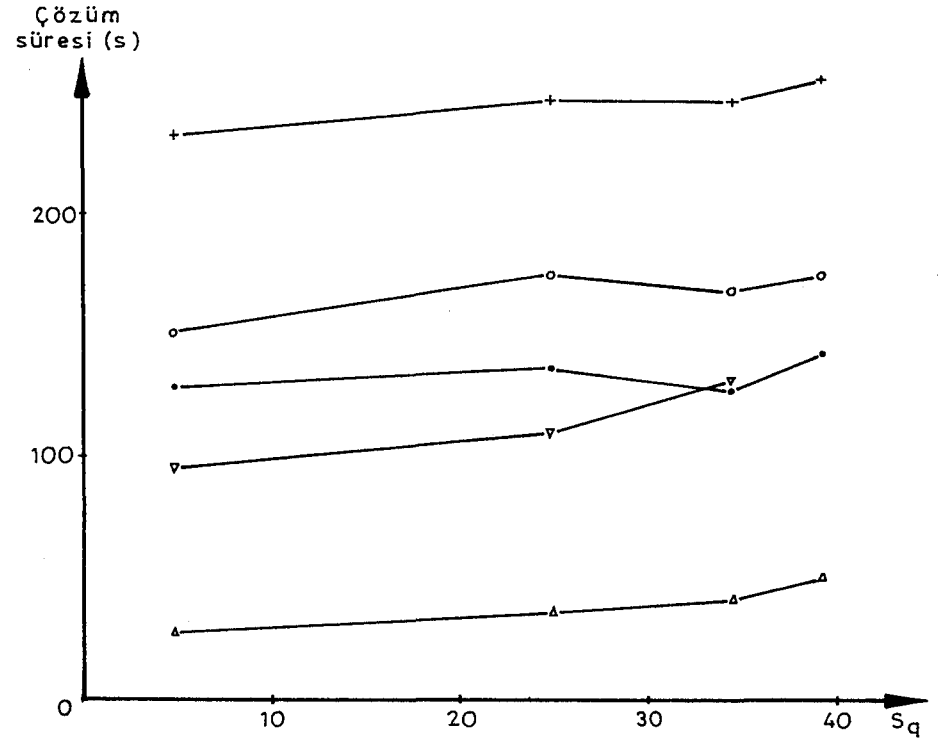
Bu değerlendirmeler sonucunda, En Hızlı İniş, Fletcher-Reeves ve Davidon-Fletcher-Powell yöntemlerinde başlangıç Q değerlerinin sıfırdan farklı değerler olarak atanmasının çalışma performansı üzerinde çok büyük değişiklikler yaratmadığı söylenebilmektedir. Newton yönteminde ise Hessien matrisinin inversinin var olabilmesi için, başlangıç değer-

Çizelge 7.4 : Örnek Şebeke-4 için, değişik başlangıç Q değerleri ile elde edilen çalışma performansları.

Kol	Direnç (Murgue)	Q_1^o	Q_2^o	Q_3^o	Q_4^o	Optimum Çözüm
1	1,1	0	10	30	70	77,8016
2	10000	0	1	4	4	2,6112
3	0,5	0	9	26	66	75,1904
4	14,8	0	5	16	50	52,8715
5	21	0	4	10	16	22,3189
6	252,7	0	1	6	10	9,7436
7	47,6	0	2	10	14	12,3548
8	7,9	0	4	10	40	43,1279
9	6,6	0	2	4	12	15,9242
10	13,6	0	4	16	27	30,5005
11	215	0	2	4	8	11,3184
12	19,2	0	2	6	28	27,2038
13	257,9	0	2	6	9	13,5399
14	19,2	0	6	20	35	41,8189
15	244	0	1	3	14	14,1117
16	45,2	0	2	6	13	18,1457
17	1000	0	1	3	14	13,0921
18	3000	0	2	4	7	8,7791
19	0,9	0	3	7	21	21,8711
20	39,4	0	7	23	49	55,9306
21	1,9	0	10	30	70	77,8016
Tahminin standard hatası		38,75	34,04	24,61	4,65	0
H.C.	İterasyon	14	11	8	5	
	Çözüm süresi	52	44	36	28	
E.H.İ.	İterasyon	35	34	34	33	
	Çözüm süresi	263	254	252	238	
F-R	İterasyon	15	13	14	13	
	Çözüm süresi	148	132	142	132	
D-F-P	İterasyon	13	12	13	11	
	Çözüm süresi	181	173	184	152	
Newton	İterasyon	-	7	6	4	
	Çözüm süresi	-	168	145	97	



Şekil 7,10 : Örnek Şebeke-4 için başlangıç Q değerleri ile iterasyon sayısının değişimi.



Şekil 7.11 : Örnek Şebeke-4 için başlangıç Q değerleri ile çözüm süresinin değişimi.

leri sıfır olarak atanamamakta, akımın korunumu ilkesini gerçekleyecek şekilde sıfırdan farklı başlangıç değerlerinin atanması zorunlu olmaktadır.

Newton yöntemi dışındaki kısıtsız optimizasyon tekniklerinde, ilk Q değerlerinin sıfır olarak veya sıfırdan farklı değerler olarak atanmasının çalışma performansını fazla etkilememesi, bu teknikler için avantajlı bir durum yaratmaktadır. Başlangıç Q değerlerini sıfır olarak atamakla, bu değerler için akımın korunumu ilkesini gerçekleyecek şekilde mantıki değerlerin verilmesi sorunundan kurtulunmaktadır.

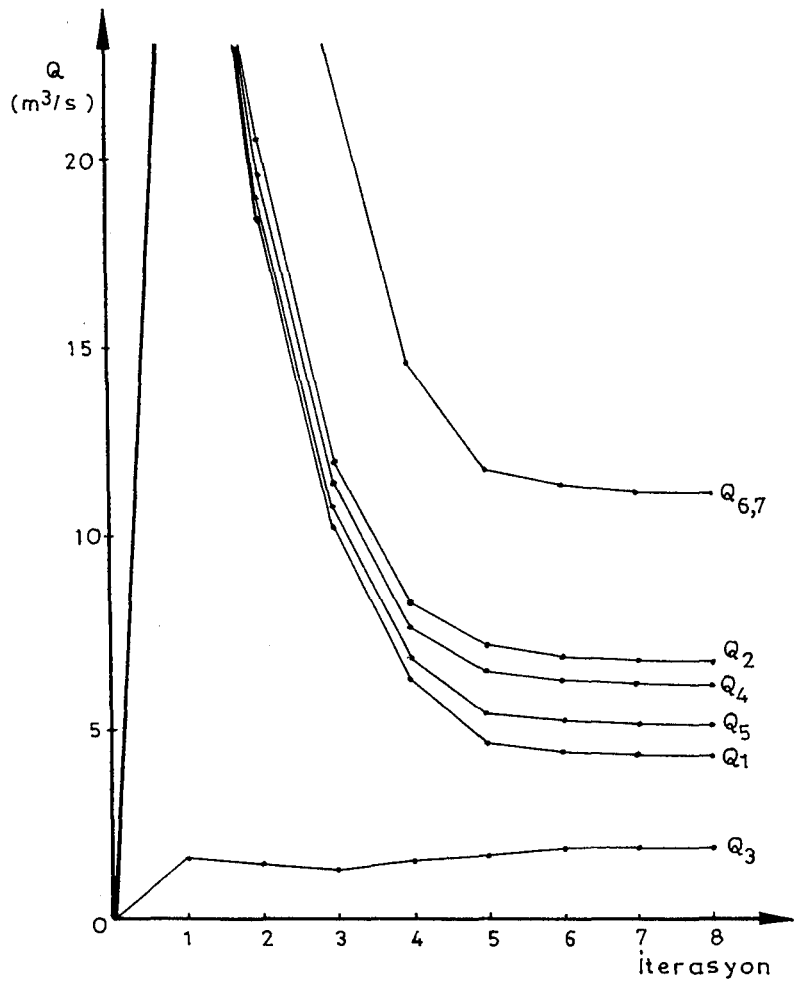
7.7. Optimum Çözüme Yaklaşım

Havalandırma şebekelerinin analizinde kullanılan çözüm tekniklerinin sıfır başlangıç mümkün çözümlerinden (Newton yöntemi dışında), optimum çözüme yaklaşım davranışlarını karşılaştırmak amacıyla, her iterasyonda kollardaki Q^k değerleri bilgisayar çıktısı olarak bastırılmıştır.

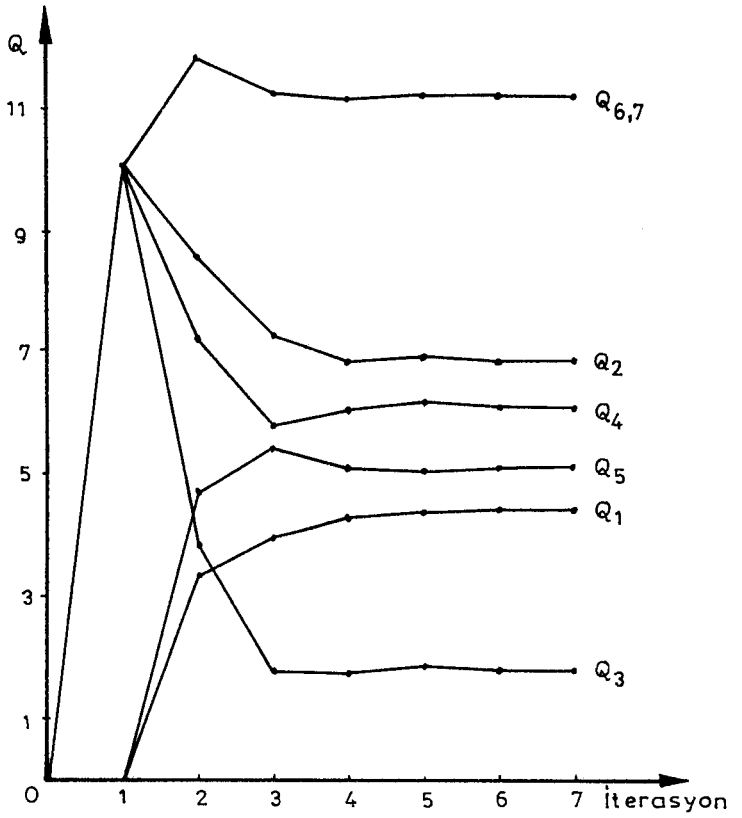
Bu araştırma için, kollardaki hava miktarlarının değişimini daha kolay izleyebilmek bakımından, kol sayısı en az olan Örnek Şebeke-1 ele alınmıştır. Bu şebeke için çözüm tekniklerinin uygulanmasında, kollardaki Q değerlerinin optimum çözüme yaklaşımları Şekil 7,12-Şekil 7,16' da verilmektedir.

Optimum çözüm seti ile her iterasyonda bastırılan Q değerleri arasında tahminin standard hataları hesaplanarak, her adımdaki optimum çözüme yaklaşım miktarı bu parametre ile ifade edilmiştir.

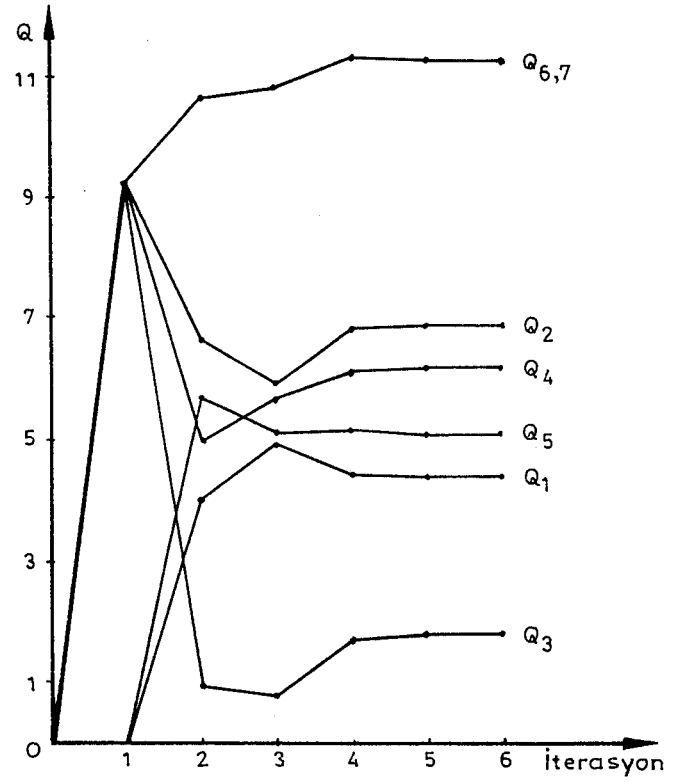
Örnek Şebeke-1 için her iterasyonda ulaşılan mümkün çözüm setleri 6. Bölümdeki çizelgelerde verilmişti. Tüm örnek şebekeler için her iterasyondaki tahminin standard hataları ise Çizelge 7,5-Çizelge 7,7'de verilmiştir.



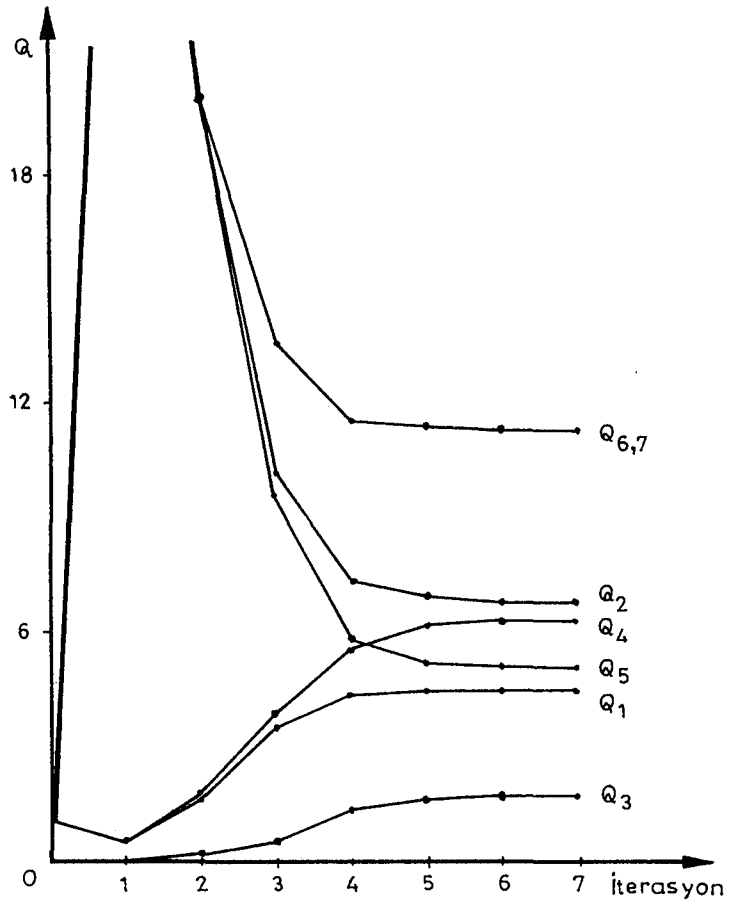
Şekil 7.12 : Hardy Cross yöntemi ile Örnek Şebeke-1 için optimum çözüme yaklaşım.



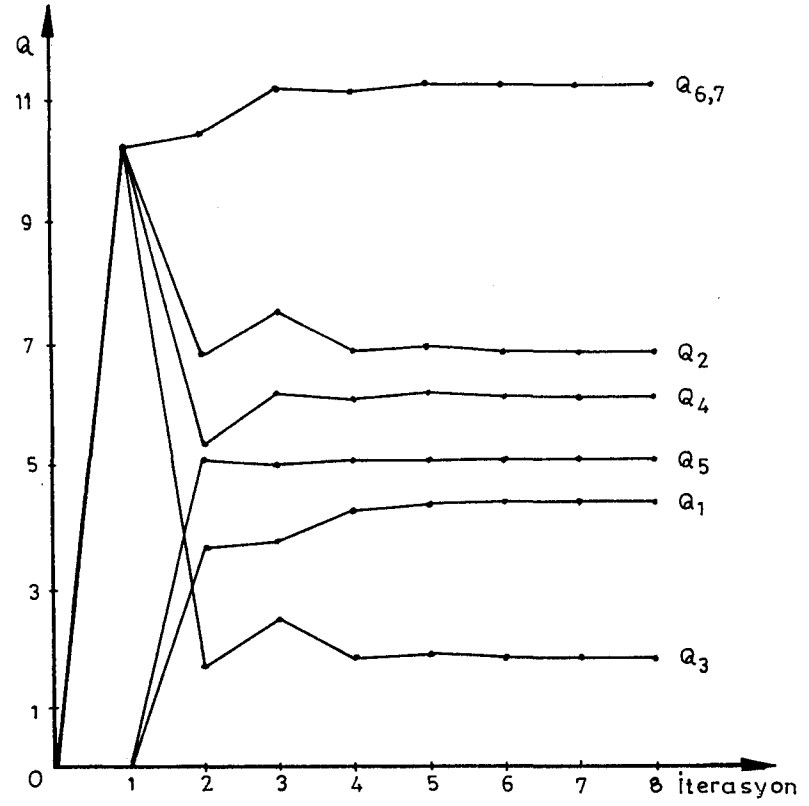
Şekil 7.13 : Fletcher-Reeves yöntemi ile Örnek Şebeke-1 için optimum çözüme yaklaşım.



Şekil 7.14 : Davidon-Fletcher-Powell yöntemi ile Örnek Şebeke-1 için optimum çözüme yaklaşım.



Şekil 7.15 : Newton yöntemi ile Örnek Şebeke-1 için optimum çözüme yaklaşım.



Şekil 7.16 : En Hızlı İniş yöntemi ile Örnek Şebeke-1 için optimum çözüme yaklaşım.

Çizelge 7.5 : Örnek Şebeke-1 için her iterasyondaki Q değerlerinin tahminin standard hataları.

İterasyon	H.C.	E.H.İ.	F-R	D-F-P	Newton
1	39,5482	4,5571	4,5098	4,2081	22,8957
2	18,0893	0,6335	1,2053	0,7716	8,7379
3	7,1163	0,4416	0,2946	0,6581	2,7321
4	2,0777	0,0671	0,0439	0,0622	0,4411
5	0,3608	0,0452	0,0254	0,0001	0,0144
6	0,0357	0,0086	0,0001	-	0,0001
7	0,0026	0,0031	-	-	-
8	-	-	-	-	-

Çizelge 7.6 : Örnek Şebeke-2 için her iterasyondaki Q değerlerinin tahminin standard hataları.

İterasyon	H.C.	E.H.İ.	F-R	D-F-P	Newton
1	212,31	2,7761	2,7832	2,7811	51,8459
2	122,78	1,1219	1,5869	1,1245	23,7359
3	72,154	0,928	0,9457	0,7212	10,0632
4	43,298	0,642	0,6484	0,0473	3,7651
5	26,681	0,603	0,6011	0,0138	1,0534
6	16,772	0,412	0,1743	-	0,1253
7	10,161	0,395	0,0428		0,0021
8	3,1811	0,2738	0,0202		-
9	1,0695	0,2537	0,0966		
10	0,1051	0,1711	0,0019		
11	0,0051	0,1571	-		
12	0,0004	0,1107			
13	-	0,0981			
14		0,0652			
15		0,0578			
16		0,0382			
17		0,0332			
18		0,0209			
19		0,0141			
20		0,0069			
21		0,0046			
22		-			

Çizelge 7.7 : Örnek Şebeke-3 için her iterasyondaki Q değerlerinin tahminin standard hataları.

İterasyon	H.C.	E.H.İ.	F-R	D-F-P	Newton
1	330,845	3,6362	3,6302	3,8465	31,1211
2	181,791	1,5085	1,3106	1,6067	13,8364
3	100,252	1,3653	0,7457	1,1054	5,4985
4	54,475	0,6439	0,2347	0,9205	1,7893
5	29,057	0,5726	0,1199	0,5652	0,3572
6	14,901	0,2906	0,0899	0,2173	0,2121
7	7,046	0,2515	0,0719	0,0158	0,0001
8	2,905	0,1398	0,0026	0,0002	-
9	1,025	0,1209	-	-	
10	0,3047	0,0831			
11	0,0811	0,0695			
12	0,0175	0,0649			
13	0,0037	0,0572			
14	0,0007	0,0174			
15	-	0,0122			
16		0,0086			
17		0,0046			
18		0,0043			
19		-			

Hardy Cross yönteminde ilk iterasyonda optimum noktanın çok uzaklarına düşülmesine karşın, sonraki iterasyonlarda hızlı bir yakınsama gözlenmekte, optimum nokta yakınlarında fazla salınım oluşmamaktadır. Bir başka deyimle, her iterasyonda optimum değere yaklaşım oranı yüksek olmaktadır.

Newton yöntemi dışındaki kısıtsız optimizasyon tekniklerinde ise daha ilk iki iterasyonda optimum noktaya, tahminin standard hatası değeri 0,6-1,6 olacak kadar yaklaşılabilmektedir. Hardy Cross yönteminde bu derecede yaklaşım ancak 5'inci-9'uncu iterasyonlarda sağlanabilmektedir.

Bu olumlu özelliklerine karşın, optimizasyon tekniklerinin daha sonraki iterasyonlarında ve özellikle optimum nokta yakınlarında çok

yavaş bir yaklaşım gözlenmektedir. En Hızlı İniş yönteminde bu özellik çok daha belirgin olmakta, Örnek Şebeke-2 için 2. iterasyonda tahminin standard hatası 1,1219 olan noktadan optimum noktaya ulaşabilmek için 20 iterasyon yapılması gerekmektedir. Bu sayı Fletcher-Reeves yönteminde 9 iterasyona, Davidon-Fletcher-Powell yönteminde ise 4 iterasyona inmektedir.Bu durum, En Hızlı İniş yönteminin optimum nokta yakınlarında zigzag çizerek ilerleme özelliğinden kaynaklanmaktadır.

Sonuç olarak, kısıtsız optimizasyon tekniklerinin ilk iterasyonlarda avantaj sağlamalarına karşın, daha sonraki iterasyonlarda yavaş bir yaklaşım gösterdikleri ve bu nedenle toplam çözüm süresi bakımından Hardy Cross yöntemine üstünlük sağlayamadıkları belirlenmiştir.

8. KOMBİNE YÖNTEMİN GELİŞTİRİLMESİ

8.1. Amaç

Havalandırma şebekelerine uygulanan kısıtsız optimizasyon tekniklerinin çözüm süresi bakımından Hardy Cross yöntemine üstünlük sağlayamadığı belirlenmiştir. Ancak bu tekniklerin sıfır başlangıç Q değerleri ile dahi, ilk birkaç iterasyonda optimum noktaya çok yaklaşabildiği, daha sonraki iterasyonlarda ise yavaş bir yakınsama sağladığı belirlenmiş, Hardy Cross yönteminde ise bunun tam tersine bir davranış gözlemlenmiştir.

Elde edilen sonuçlar ışığında, Hardy Cross ve kısıtsız optimizasyon tekniklerinin avantajlarını birleştiren kombine bir yöntemin geliştirilmesi ve böylece Hardy Cross yöntemine üstünlük sağlanabileceği düşünülmüştür.

Oluşturulacak Kombine yöntemde ilk birkaç iterasyonun kısıtsız optimizasyon tekniklerinden birisi ile yapılarak optimum noktaya yaklaşılması, daha sonraki iterasyonlarda Hardy Cross tekniğine geçilerek optimum nokta civarındaki yakınsamanın hızlandırılması planlanmıştır.

8.2. Kombine Yöntemde İzlenen Algoritma

Kısıtsız optimizasyon tekniklerinden birisini ve Hardy Cross yöntemini aynı algoritma içinde kullanan bir yöntemin, bilgisayar belleğinde daha fazla yer kullanacağı açıktır. Kullanılan bellek hacmını düşük tutabilmek için ilk iterasyonların, diğerlerine oranla daha az yer kaplayan En Hızlı İniş yöntemi ile yapılması uygun olacaktır. Optimum çözüme ulaşana dek yapılan tekrarlama sayısının fazla olmasına karşın, bu yöntemde bir tek iterasyon için harcanan sürenin diğer tek-

niklere oranla daha az olması, adı geçen yöntemin bir başka avantajını oluşturmaktadır.

Havalandırma şebekelerinin Kombine yöntem ile analizinde aşağıda verilen algoritma geliştirilerek, bilgisayar kodlamasında izlenmiştir.

1. Adım : İstenen hassasiyet değeri (ε), En Hızlı İniş yönteminin uygulanacağı tekrarlama sayısı (t) seçilir, k=0 alınır.

2. Adım : En Hızlı İniş yöntemi uygulanır.

3. Adım : k=k+1 yapılır,
k<t ise 2. adıma dönülür.

4. Adım : Başlangıç noktası olarak Q^t alınır.

5. Adım : Hardy Cross tekniği uygulanır.

6. Adım : k=k+1 yapılır,
$|Q^k-Q^{k-1}|<\varepsilon$ ise işlem bitirilir, tersine durumda 4. adıma dönülür. İşlem bitirildiğinde ulaşılan nokta, optimum çözüm setini verecektir.

Algoritmanın 2. adımında En Hızlı İniş yöntemi uygulanırken, negatif gradyan vektörü yönündeki ilerleme miktarını belirlemek için doğrusal arama yapılması gerekmektedir. Bu adımda amaç, hassas bir sonuç almaktan çok, optimum noktaya kısa sürede yaklaşmak olduğundan, doğrusal arama işleminde tek iterasyonlu kareli interpolasyon yapılması yeterli görülmüştür. Bu düzenleme ile, daha az hassas başlangıç Q^t değerleriyle, ancak çok daha kısa sürede Hardy Cross tekniğine geçilmesi sağlanmıştır.

8.3. Bilgisayar Programı

Kombine yöntem için, verilen algoritmayı izleyen bir bilgisayar programı yazılarak örnek şebekeler için denenmiş ve işlerliği kanıtlanmıştır.

Programın ilk bölümünde En Hızlı İniş yöntemi uygulanmakta, ikinci bölümde ise Hardy Cross tekniğindeki düzeltme işlemleri yapılmaktadır. Programdaki sabit ve değişkenlerin isimleri, daha önce verilmiş olan sabit ve değişken isimleri ile uyumlu olarak kullanılmıştır.

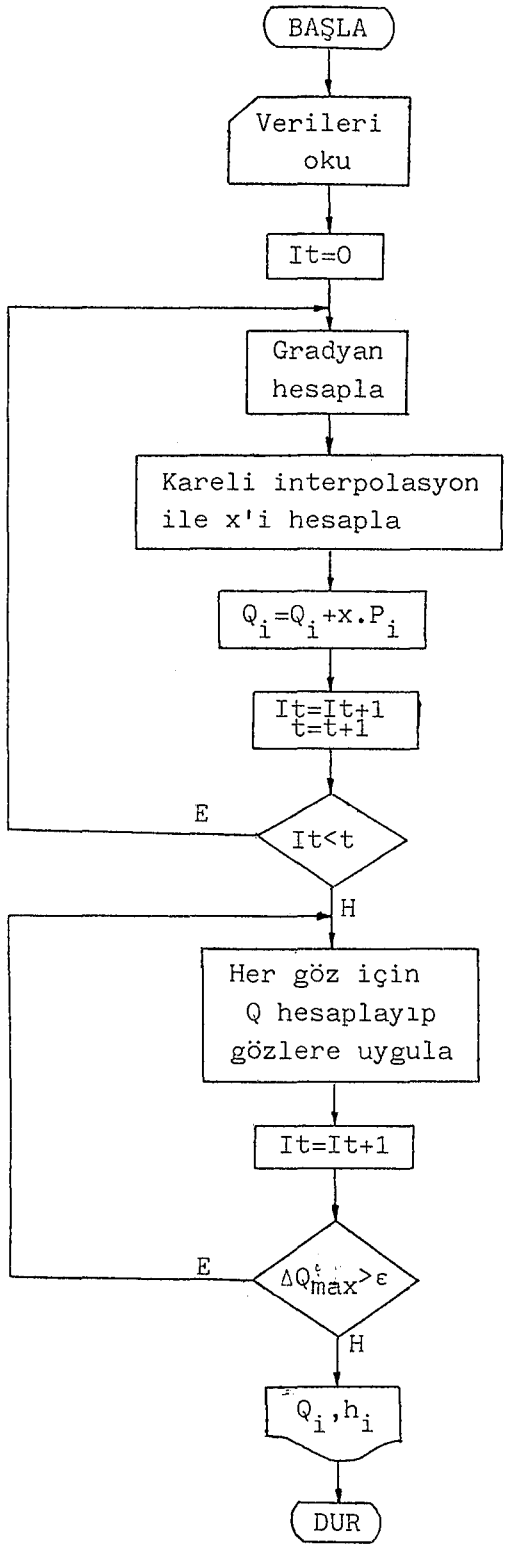
Bilgisayar programının yapısında bütünlük sağlamak amacıyla, gradyan hesaplama ve doğrusal arama alt programları ana program içine sokulmuş, alt program olarak sadece fonksiyon değerini hesaplama alt programı kullanılmıştır. Gradyan hesaplama tekniği En Hızlı İniş yönteminde kullanıldığı şekilde uygulanmış, kareli interpolasyon işlemi ise tek iterasyonlu olarak uygulanmıştır.

BASIC diliyle yazılan program EK-11'de, genelleştirilmiş akış şeması ise Şekil 8,1'de verilmiştir.

Algoritmanın ikinci adımında uygulanan En Hızlı İniş yönteminin tekrarlanma sayısı, yöntemin toplam çözüm süresini büyük oranda etkilemektedir. Optimum tekrarlama sayısını belirlemek için örnek şebekeler değişik tekrarlama sayıları ile çözümlenmiş ve yöntemin çalışma performansındaki değişiklikler incelenmiştir. Bu araştırmanın sonuçları Çizelge 8,1, Çizelge 8,2, Şekil 8,2 ve Şekil 8,3 de verilmiştir.

En Hızlı İniş yönteminin tekrarlanma sayısı (t) arttıkça, Hardy Cross tekniği için başlangıç Q^t değerleri optimum noktaya yaklaşmaktadır (Çizelge 8,1). Ancak tekrarlama sayısının bu artışı ile toplam iterasyon sayısı ve çözüm süresi de artma eğilimi göstermektedir. Çözüm süresini enküçük yapabilmek için bu iki eğilimin dengelenmesi gerekmektedir.

Şekil 8,2 ve Şekil 8,3'ün incelenmesinden, çözüm süresini enküçük yapan optimum t değerinin 2 olduğu anlaşılmaktadır. Bu nedenle, Kombine yöntemin uygulanmasında En Hızlı İniş tekniğinin tekrarlanma sayısı t=2 olarak kullanılmış, algoritmanın 3. adımında k=2 olduğunda, algoritmanın 4. adımına geçilmiştir.



Şekil 8.1 : Kombine yöntemin bilgisayar programının genelleştirilmiş akış şeması.

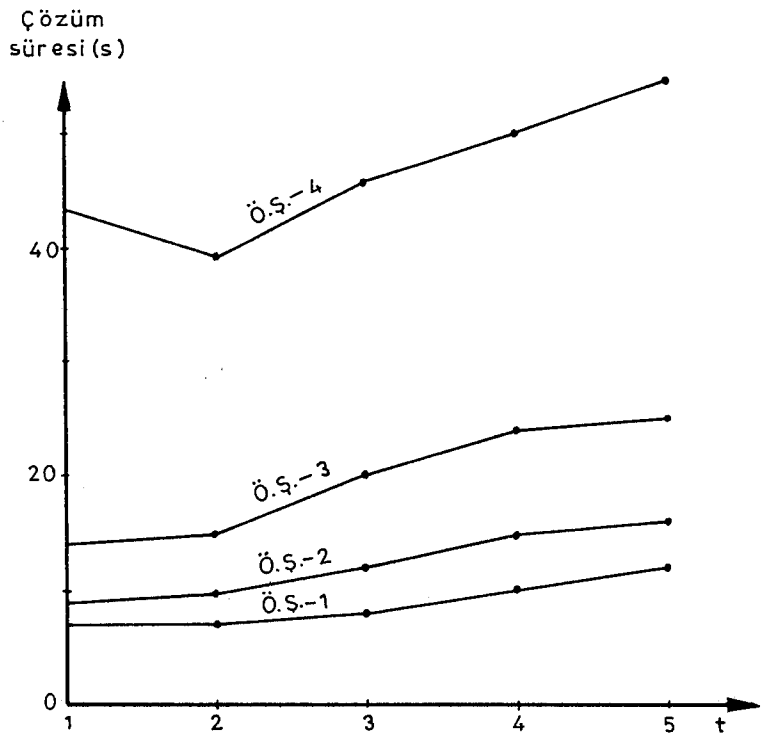
Çizelge 8.1 : Değişik tekrarlama sayıları (t) ile, Q^t değerlerinin değişimi (Örnek Şebeke-1 için).

Tekrarlama sayısı (t)	1	2	3	4	5	Optimum çözüm
Q^t	0	0,5165	2,3567	2,5908	3,7332	4,3823
	8,4577	10,1628	7,4159	8,4448	7,0157	6,8818
	8,4577	9,4311	4,1622	4,9632	2,3189	1,8031
	8,4577	9,9476	6,5188	7,5541	6,0522	6,1853
	0	0,7317	3,2537	3,4816	4,6967	5,0787
	8,4577	10,6793	9,7726	11,0356	10,7489	11,2641
	8,4577	10,6793	9,7726	11,0356	10,7489	11,2641
S.H.	4,1121	4,0993	1,5966	1,6974	0,4471	0

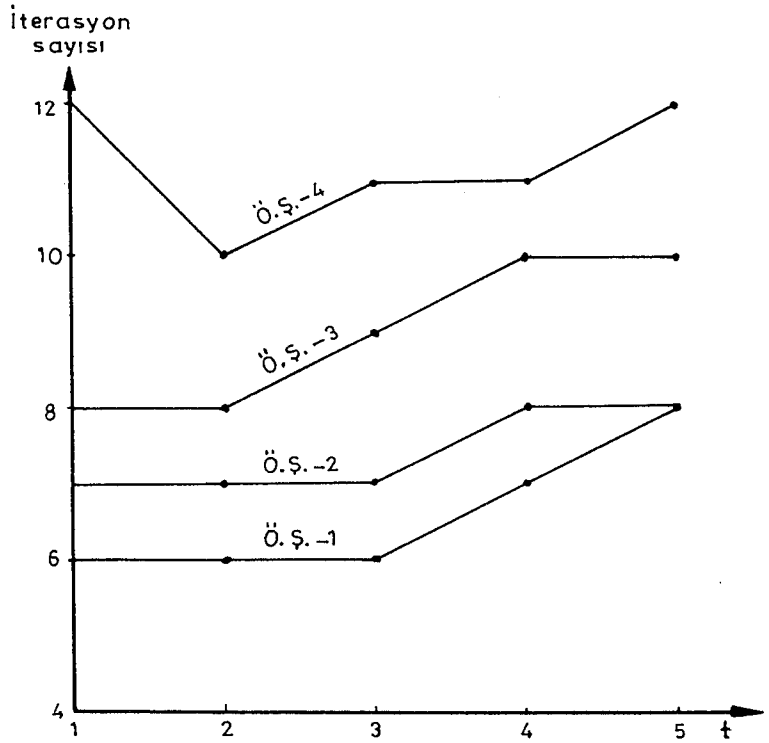
Çizelge 8.2 : Örnek şebekeler için değişik tekrarlama sayıları ile optimum çözüme ulaşma özellikleri.

Tekrarlama sayısı (t)		1	2	3	4	5
Şebeke-1	İ.S.	6	6	6	7	8
	Ç.S.	7	7	8	10	12
	S.H.	4,1121	4,0993	1,5966	1,6974	0,4471
Şebeke-2	İ.S.	7	7	7	8	8
	Ç.S.	9	10	12	15	16
	S.H.	2,5882	2,5615	1,3552	1,3949	0,8354
Şebeke-3	İ.S.	8	8	9	10	10
	Ç.S.	14	16	20	24	25
	S.H.	3,3349	1,8594	1,0291	0,9168	0,5443
Şebeke-4	İ.S.	12	10	11	11	12
	Ç.S.	43	39	46	50	55
	S.H.	29,981	7,313	7,1511	4,5779	4,2872

İ.S. : İterasyon sayısı,
Ç.S. : Çözüm süresi,
S.H. : Tahminin standard hatası.



Şekil 8.2 : Tekrarlama sayısı ile toplam çözüm süresinin değişimi (Ö.Ş. : Örnek şebeke).



Şekil 8.3 : Tekrarlama sayısı ile iterasyon sayısının değişimi (Ö.Ş. : Örnek şebeke).

8.4. Hardy Cross Yöntemi İle Karşılaştırma

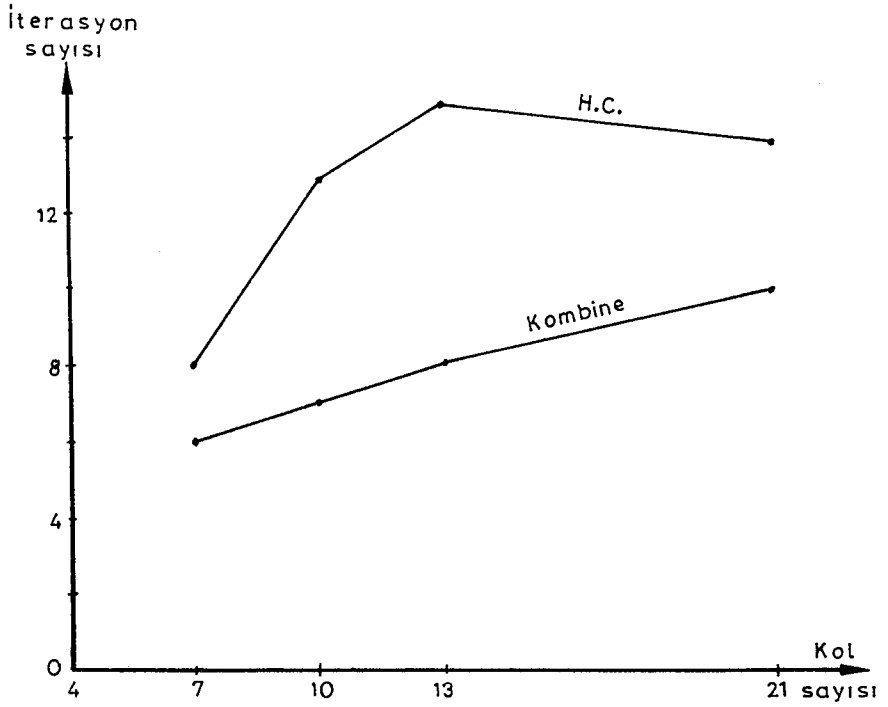
Geliştirilen Kombine yöntemin çalışma performansını yorumlamak amacıyla, yöntem örnek şebekelere uygulanmış, elde edilen sonuçlar Hardy Cross tekniği sonuçlarıyla karşılaştırılmıştır. Her iki yöntemin optimum çözüme ulaşma özellikleri Çizelge 8,3'de verilmiştir.

Çizelge 8.3 : Hardy Cros ve Kombine yöntemlerin örnek şebekeler için verdiği sonuçlar.

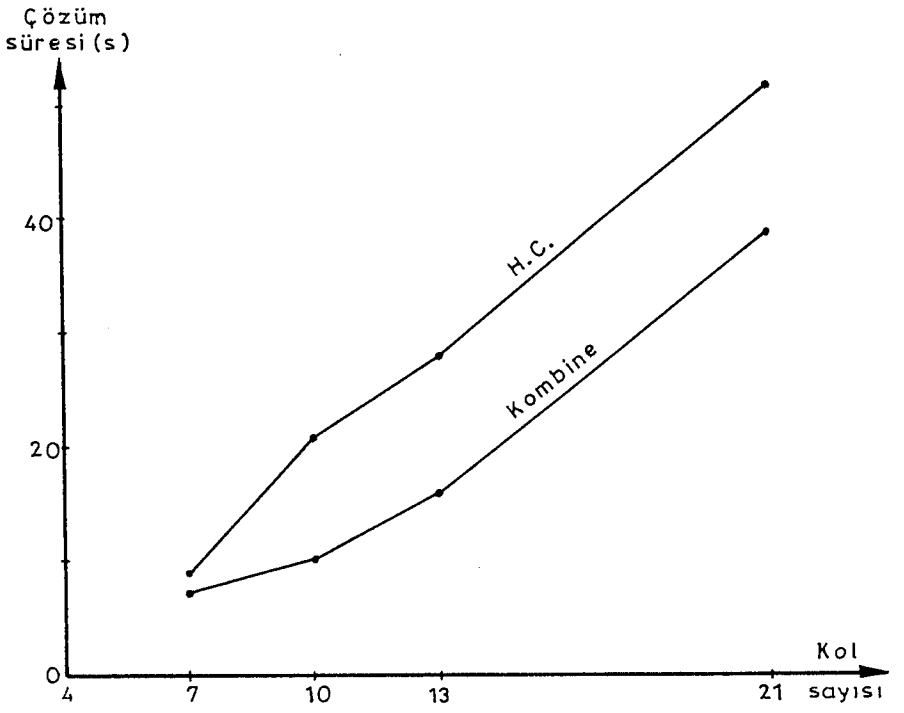
		H.C.	Kombine
Örnek Şebeke-1	İterasyon sayısı Çözüm süresi (s) Bellek hacmı (B)	8 9 1766	6 7 3685
Örnek Şebeke-2	İterasyon sayısı Çözüm süresi (s) Bellek hacmı (B)	13 21 1798	7 10 3711
Örnek Şebeke-3	İterasyon sayısı Çözüm süresi (s) Bellek hacmı (B)	15 28 1843	8 16 3722
Örnek Şebeke-4	İterasyon sayısı Çözüm süresi (s) Bellek hacmı (B)	14 52 2123	10 39 3777

Her iki yöntemin çözüm süresi, iterasyon sayısı ve kullanılan bellek hacmı bakımından grafiksel karşılaştırmaları ise Şekil 8,4, Şekil 8,5 ve Şekil 8,6' da verilmiştir.

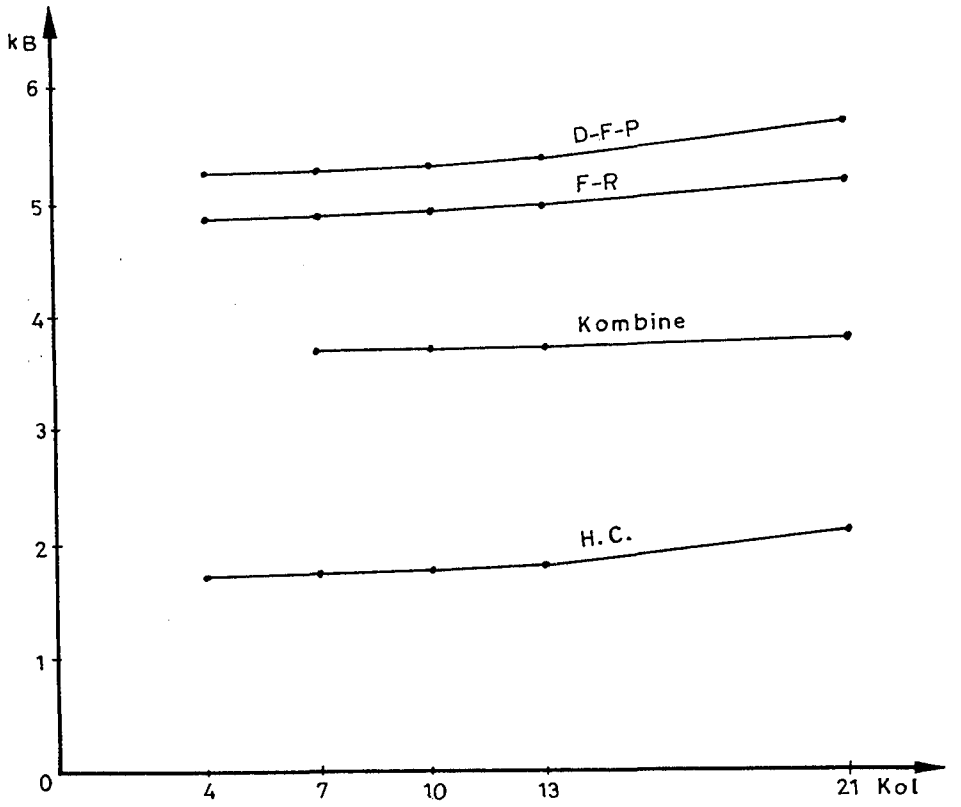
Çizelge ve şekillerin incelenmesinden, geliştirilen Kombine yöntemin gerek iterasyon sayısı, gerekse çözüm süresi bakımından Hardy Cross tekniğine üstünlük sağladığı anlaşılmaktadır. 21 kollu Örnek Şebeke-4 için Kombine yöntem, iterasyon sayısını 14'den 10'a, çözüm süresini ise 52 saniyeden 39 saniyeye düşürmektedir.



Şekil 8.4 : Kombine yöntem ile Hardy Cross tekniğinin iterasyon sayısı açısından karşılaştırılması.



Şekil 8.5 : Kombine yöntem ile Hardy Cross tekniğinin çözüm süresi açısından karşılaştırılması.



Şekil 8.6 : Kombine yöntemin, Hardy Cross tekniği ile kullanılan bellek hacmı açısından karşılaştırılması.

Elde edilen bu sonuçların her iki yöntemde de sıfır başlangıç mümkün çözümü ile alındığı unutulmamalıdır. Optimum noktaya yakın başlangıç Q değerleri atandığında Hardy Cross tekniği, sonuca daha kısa sürede ulaşabilecektir (Çizelge 7,4). Ancak, Örnek Şebeke-4 üzerinde yapılan çalışmalarda, optimum noktadan uzak atamalar yapıldığında veya kollardaki akım yönlerinin gerçek yönlerden çok farklı olarak atandığında, bu üstünlüğün kaybolduğu, hatta çözüm süresinin sıfır başlangıç değerleri atanmasından daha da fazla olabildiği belirlenmiştir.

Kombine yöntemde ise başlangıç Q değerleri sıfır olarak alınmakta, optimum noktaya çabuk bir şekilde yaklaşılmaktadır. Böylece Hardy Cross tekniğindeki, başlangıç Q değerlerini Kirchoff'un akımın korunumu ilkesine uygun olarak atamak zahmetinden de kurtulunmuş olunmaktadır.

Kombine yöntemin tek sakıncası, bilgisayar belleğinde daha fazla yer kullanması olarak belirmektedir. Ancak yöntem, bu açıdan da D-F-P ve F-R yöntemlerine oranla avantajlı olmaktadır (Şekil 8,6).

Bilgisayar teknolojisindeki küçük hacımlı, büyük kapasiteli bilgisayarlara doğru olan gelişmeler, fazla bellek kullanma sakıncasını önemsiz kılmaktadır. Günümüzde 128 kB kapasiteli mikro bilgisayarların ev bilgisayarı olarak kullanılması bu görüşü doğrulamaktadır.

Sonuç olarak, geliştirilen Kombine yöntemin Hardy Cross yöntemine oranla çözüme daha kısa sürede ulaştığı, başlangıç Q değerlerinin atanmasını gerektirmediği ve havalandırma şebekelerinin analizinde büyük kolaylıklar getirdiği gösterilmiş olmaktadır.

8.5. Genel Amaçlı Bilgisayar Programı

Önceki bölümlerde, ele alınan kısıtsız optimizasyon teknikleri için yazılan bilgisayar programlarında çözüm tekniklerinin karşılaştırılması amaçlandığından, göz dizileri ve vantilatör karakteristik eğrisinin katsayıları gibi gerekli parametreler hazır veri olarak okutulmuştu.

Üstünlüğü kanıtlanan Kombine yönteme daha pratik işlerlik kazandırmak amacıyla, göz dizilerinin seçimi, vantilatör katsayılarının hesaplanması ve doğal hava akımının etkisinin analizini de içeren genel amaçlı bir bilgisayar programının hazırlanması yoluna gidilmiştir. EK-12'de verilen program, bir ana ve üç alt programdan oluşmaktadır.

8.5.1. Vantilatör katsayıları hesabı alt programı

Vantilatör katsayılarının bilinmesi durumunda, bu değerler veri olarak okutulmakta, tersine durumda ise bir alt programda hesaplatılmaktadır.

Vantilatör katsayılarının belirlenmesinde, vantilatör karakteristik eğrisinin konkav bir parabol gösterdiği ve ikinci dereceden bir po-

linomla ifade edilebileceği kabul edilmiş, hesaplamada eğrisel regresyon tekniği kullanılmıştır.

Katsayıları aranan parabol denklemi,

$$h = A + B.Q + C.Q^2 \qquad (8\text{-}1)$$

şeklinde olduğundan en küçük kareler denklemi,

$$S = \sum_{i=1}^{n} (A + B.Q_i + C.Q_i^2 - h_i)^2 \qquad (8\text{-}2)$$

şeklinde yazılabilir. S fonksiyonunun sırasıyla A, B, C katsayılarına göre kısmi türevleri alınıp sıfıra eşitlendiğinde oluşan denklem sistemi matris notasyonu ile ifade edilirse,

$$\begin{bmatrix} N & \Sigma Q_i & \Sigma Q_i^2 \\ \Sigma Q_i & \Sigma Q_i^2 & \Sigma Q_i^3 \\ \Sigma Q_i^2 & \Sigma Q_i^3 & \Sigma Q_i^4 \end{bmatrix} \cdot \begin{bmatrix} A \\ B \\ C \end{bmatrix} = \begin{bmatrix} \Sigma h_i \\ \Sigma (Q_i . h_i) \\ \Sigma (Q_i^2 . h_i) \end{bmatrix} \qquad (8\text{-}3)$$

$$M \,.\, X = F \qquad (8\text{-}4)$$

matris çarpımı elde edilir. Vantilatörün bilinen hava miktarı-yük değerlerinden yararlanarak,

$$Q = \begin{bmatrix} 1 & Q_1 & Q_1^2 \\ 1 & Q_2 & Q_2^2 \\ \vdots & \vdots & \vdots \\ 1 & Q_n & Q_n^2 \end{bmatrix} \quad , \quad H = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_n \end{bmatrix} \qquad (8\text{-}5)$$

matrisleri oluşturulursa,

$$\begin{aligned} M &= Q^T . Q \\ V &= Q^T . H \end{aligned} \qquad (8\text{-}6)$$

olduğu görülür. Bu durumda X katsayılar matrisi için,

$$X = (Q^T . Q)^{-1} . Q^T . H \qquad (8-7)$$

eşitliği elde edilir (Barkana ve Akgün, 1983).

Bu eşitlikte belirtilen matris işlemleri yapıldığında ulaşılan (3X1) boyutlu katsayılar matrisinin elemanları, aranan A, B, C katsayıları olmaktadır.

Vantilatör katsayıları hesaplama alt programında (8-7) ifadesindeki matris işlemleri yapılarak katsayılar hesaplandıktan sonra, belirlenen eğrisel ilişkinin korelasyon katsayısı da hesaplanarak bilgisayar çıktısı olarak basılmaktadır.

8.5.2. Göz seçimi alt programı

Yazılan genel amaçlı bilgisayar programında göz dizileri bir alt programda bilgisayara seçtirilmektedir. Bu işlem için her kolun başlangıç ve bitiş kavşaklarının bilgisayara okutulması gerekmektedir.

Göz seçim alt programında, Wang (1982) tarafından, FORTRAN IV kodunda verilen alt program bazı değişikliklerle aynen kullanılmıştır.

8.5.3. Doğal havalandırma alt programı

Doğal hava akımının etkisinin analize sokulması istendiğinde, bir alt programda gerekli hesaplamalar yaptırılmaktadır. Bu analiz için, şebekedeki her kavşağın kotunun ve bu noktalarda ölçülmüş hava sıcaklığı değerlerinin bilgisayara okutulması gerekir.

Bir koldaki doğal hava basıncı,

$$h_d = \frac{\delta_a + \delta_b}{2} . (h_b - h_a) \quad , \quad (\text{mm ss}) \qquad (8-8)$$

eşitliğinden hesaplanır. Hava yoğunluğunun saptanmasında ise,

$$\delta = \frac{10,233 - 1,25.h_a}{29,4.T} \qquad (8-9)$$

ifadesi kullanılır (Güyagüler, 1979). Bu ifadelerde;

δ_a, δ_b : Başlangıç ve bitiş kavşaklarında hava yoğunluğu (kg/m^3),

h_a, h_b : Başlangıç ve bitiş kavşaklarının kotları (m),

T : Mutlak sıcaklıktır (273+t°C).

Doğal havalandırma alt programında her koldaki doğal hava basınçları (8-8) ifadesi kullanılarak hesaplandıktan sonra, gözlerdeki doğal hava basınçları hesaplanmakta ve bilgisayar çıktısı olarak yazdırılmaktadır. Hesaplanan bu değerler, daha sonraki iterasyon işlemlerinde sabit basınç kaynağı olarak hesaba katılmaktadır.

8.5.4. Verilerin okutulması

Havalandırma şebekelerinin analizinde Kombine yöntemi kullanan genel amaçlı bilgisayar programının hatasız bir şekilde çalışabilmesi için, gerekli verilerin okutulmasında aşağıdaki algoritma izlenmelidir.

1. Adım : Şebekedeki kol sayısı (Kol), kavşak sayısı (Kav), vantilatör sayısı (Vs) okutulur.

2. Adım : Şebekede vantilatör yoksa 5. adıma atlanır.

3. Adım : a) Vantilatör katsayıları biliniyorsa;
A,B,C katsayıları ve vantilatörün bulunduğu kol numarası (Vk) okutulur.

b) Vantilatör katsayılarının hesaplanması isteniyorsa;
A,B,C katsayıları sıfır olarak verilir, vantilatörün bulunduğu kol numarası (Vk) okutulur.
Katsayıları aranan vantilatör için bilinen veri sayısı (Veri(I)), ardından önce vantilatörün hava miktarı değerleri (Qv(I)), daha sonra da yük değerleri (Hv(I)) verilir.

4. Adım : 2. ve 3. adımlar şebekedeki vantilatör sayısı kadar tekrarlanır.

5. Adım : Sırasıyla her kolun başlangıç kavşak numarası, bitiş kavşağı numarası ve direnç değerleri okutulur. K1(I),K2(I),R(I).

6. Adım : Doğal hava basıncı etkisinin analize sokulması isteniyorsa E, istenmiyorsa H verilir. İkinci durumda 8. adıma atlanır.

7. Adım : Her kolun başlangıç kavşağının kot ve hava sıcaklığı değerleri okutulur. Kot(K1(I)), Sıc(K1(I)), (I=1,2,...,Kol).

8. Adım : Okutma işlemleri bitirilir, program çalıştırılır.

Gerekli veriler doğru bir şekilde verilip, program çalıştırıldığında aşağıdaki bilgiler çıktı olarak alınmaktadır:

a) Vantilatör katsayıları ve korelasyon katsayısı,
b) Seçilen göz dizileri,
c) Gözlerdeki doğal hava basıncı değerleri,
d) Kollardaki hava dağılımı,
e) İterasyon sayısı.

EK-12'de verilen genel amaçlı bilgisayar programı Kombine yöntemi kullanarak, havalandırma şebekelerini hızlı bir şekilde çözümlemekte ve kollardaki hava dağılımı değerlerini vermektedir. Çözüm süresinin kısa olması, göz seçim işlemini otomatik olarak yapması, vantilatör karakteristik eğrisinin katsayılarının bilinmemesi durumunda, bu katsayıları hesaplaması, uygulamacı maden mühendisine büyük kolaylıklar sağlamaktadır.

9. SONUÇLAR

Havalandırma şebekelerinin kısıtsız optimizasyon teknikleri ile çözümlenebilirliğinin incelendiği bu çalışmada elde edilen sonuçlar şu şekilde sıralanabilir :

1-) Havalandırma şebekelerinin modellenmesi ile oluşturulan matematiksel model havalandırma şebekesinde tüketilen hava gücünün üçte birini ifade etmekte ve çok değişkenli, doğrusal olmayan, kısıtsız bir model yapısında olmaktadır. Problem, şebekedeki güç tüketimini enküçükleyen hava miktarı değerlerinin hesaplanması olarak şekillenmektedir.

2-) Kısıtsız modelin amaç fonksiyonu çok değişkenli, doğrusal olmayan, tam konveks bir fonksiyon olduğundan, kısıtsız optimizasyon teknikleri bu modelin çözümünde kullanılabilir. Modelin yapısındaki değişkenler, havalandırma şebekesinin ana kollarındaki Q değerleri olduğundan, yapılan iterasyon işlemleri sonucunda ulaşılan optimum çözüm seti, havalandırma şebekesindeki güç tüketimini enküçük kılan, ana kollardakı hava dağılımı değerlerini vermektedir. Tali kollardaki hava miktarları ise, göz tekniğini kullanarak akımın korunumu ilkesini sisteme uygulamak suretiyle hesaplanabilmektedir.

3-) Amaç fonksiyonunun türevini kullanan kısıtsız optimizasyon teknikleri arasından, ele alınan En Hızlı İniş, Fletcher-Reeves+ Newton ve Davidon-Fletcher-Powell yöntemleri ile havalandırma şebekeleri için oluşturulan model çözümlenebilmektedir. Bu teknikleri kullanarak hesaplanan hava miktarı değerleri, Hardy Cross yöntemi ile elde edilen değerlerle kabul edilebilecek bir yakınlık göstermektedir. Örnek şebekeler üzerinde yapılan uygulamalar, havalandırma şebekelerinin kısıtsız optimizasyon teknikleri ile çözümlenebileceğini kanıtlamaktadır.

4-) Kısıtsız optimizasyon tekniklerinin uygulanmasında, her iterasyondaki işlemlerin yoğunluğu elle çözümü olanaksız kılmakta, sayısal bilgisayarların kullanılmasını gerektirmektedir.

5-) Kısıtsız optimizasyon tekniklerinde matris ve vektör işlemlerinin yoğun olması, her iterasyonda doğrusal arama yapılması ve hesaplanan değerlerin bellekte depolanma zorunluluğu bulunmasından dolayı, bu teknikler bilgisayar belleğinde daha fazla yer kullanmaktadır.

6-) Göz dizilerinin seçimi sırasında, büyük dirençli kolların ana kol olarak alınması ve tali kolların kapalı bir göz oluşturmaması kuralı, Hardy Cross yönteminde iterasyon sayısını ve çözüm süresini büyük oranda azaltmaktadır. Göz seçiminde bu kuralı uygulamamak Davidon-Fletcher-Powell yönteminin çalışma performansında ise olumsuz bir etki yapmamaktadır. Özellikle büyük şebekeler için oldukça zor bir işlem olan, göz dizilerinin elle seçilmesi işlemi bu yöntemde çok büyük oranda basitleşmekte, hava yolları dirençlerinin değişmesi durumunda yeni göz dizilerinin seçimini gerektirmemekte, havalandırma mühendisine büyük kolaylıklar getirmektedir.

7-) Hardy Cross yönteminin çalışma hızı üzerinde büyük etkisi olan, başlangıç Q değerlerinin sıfırdan farklı ve kollardaki hava yönlerinin gerçeğe yakın olarak atanması, Fletcher-Reeves ve Davidon-Fletcher-Powell yöntemlerinin çalışma performansları üzerinde çok büyük değişiklikler yaratmamaktadır. Başlangıç Q değerlerinin sıfır olarak atanması Hardy Cross yönteminde iterasyon sayısı ve çözüm süresini büyük oranda artırmakla birlikte, anılan kısıtsız optimizasyon tekniklerinde büyük bir etki yapmamaktadır. Bu değerleri sıfır olarak atamak havalandırma mühendisini, Kirchoff I yasasına uygun olarak kollara mantıki Q° değerlerini atamak sorunundan kurtarmaktadır.

8-) Uygulanan kısıtsız optimizasyon teknikleri iterasyon sayısı bakımından avantajlı olabilmekle birlikte, her iterasyonda yapılan işlemlerin yoğunluğundan ötürü, toplam çözüm süresi bakımından Hardy Cross yöntemine üstünlük sağlayamamaktadır. Havalandırma şebekelerinin analizinde kullanılabilecek çözüm tekniklerinin karşılaştırılmasında çözüm hızı temel ölçüt olarak alındığında, Hardy Cross yöntemi en hızlı çözüm tekniği olarak kendini göstermektedir.

9-) Hardy Cross ve Newton yöntemlerinde ilk iterasyonlarda optimum noktanın çok uzaklarına düşülmekle birlikte, sonraki iterasyonlarda optimum çözüm setine yüksek bir yakınsama hızı ile yaklaşılmaktadır. Diğer kısıtsız optimizasyon tekniklerinde ise bunun tersine bir davranış gerçeklenmekte, ilk iterasyonlarda optimum noktaya oldukça yaklaşılabilmekle birlikte, yavaş bir yakınsama hızı ile ilerlenmektedir.

10-) Bu özelliğin belirlenmesinden hareket edilerek geliştirilen Kombine yöntem, her iki avantajı da içermekte, ilk üç iterasyonda En Hızlı İniş tekniğini uygulayarak optimum noktaya yaklaşılmaktadır. Sonraki iterasyonlarda ise Hardy Cross tekniğini kullanarak yüksek bir yakınsama hızı ile optimum çözüm setine ulaşmaktadır. Kombine yöntem, Hardy Cross tekniğine oranla daha az iterasyonda ve daha düşük sürelerde sonuca gitmektedir.

11-) Kombine yöntemde başlangıç Q değerleri sıfır olarak atandığından, bu yöntemi kullanan havalandırma mühendisi, bu değerleri akımın korunumu ilkesine göre atama sorunundan kurtulunmuş olmaktadır.

12-) Kombine yöntem, Hardy Cross tekniğine oranla bilgisayar belleğinde daha fazla yer kullanmakla birlikte, günümüz bilgisayar teknolojisi için bu özellik önemli bir sakınca oluşturmamaktadır.

13-) Kombine yöntemin havalandırma şebekelerinin analizinde kullanılmasına daha pratik işlerlik kazandırmak için yazılan genel amaçlı bilgisayar

programı göz seçimini otomatik olarak yapmakta, katsayıların bilinmemesi durumunda vantilatör katsayılarını hesaplamakta, istenildiğinde doğal havalandırma basıncının etkisini de analize sokmakta ve başlangıç Q değerlerini kendiliğinden sıfır olarak atamaktadır. Kombine yöntemi kullanan program, Hardy Cross tekniğini uygulayan programlara oranla daha az iterasyonda ve daha kısa sürede çözüme ulaşmaktadır. Hazırlanan program, yeraltı havalandırma şebekelerinin analizi için, uygulamacı maden mühendisine önerilmektedir.

KAYNAKLAR DİZİNİ

Abadie, J., 1967, Nonlinear Programming, North Holland Pub. Co.

Aoki, M., 1971, Introduction to Optimization Techniques, MacMillan Pub. Co., New York.

Avriel, M., 1976, Nonlinear Programming-Analysis and Methods, Prentice-Hall Inc., Englewood Cliffs, N.J.

Ayvazoğlu, E., 1973, E.K.İ. Kozlu Bölgesi Havalandırma Problemlerinin Etüdü, İ.T.Ü. Doktora tezi, İstanbul.

Ayvazoğlu, E., 1986, Madenlerde Havalandırma ve Emniyet, İ.T.Ü. Kütüphanesi, Yayın No:12, 303 S., İstanbul.

Barkana, A., Akgün, Ö.R., 1983, BASIC Programlama ve Nümerik Hesap, Bilim Teknik Yayınevi, Eskişehir.

Bazaraa, M., Shetty, C.M., 1979, Nonlinear Programming-Theory end Algorithms, John Wiley and Sons, New York, 560 p.

Beightler, C., Phillips, D., Wilde, D., 1979, Foundations of Optimization, Prentice-Hall Inc., 483 p.

Bhamidipati, S., Procarione, J.A., 1985, Linear Analysis for the Solution of Flow Distribution Problems in Mine Ventilation Networks, 2nd US Mine Ventilation Symposium, pp. 645-654.

Bhamidipati, S., Procarione, J.A., 1986, Nonlinear Programming Technique for Analysis of Mine Ventilation Networks, Trans. Ins. Min. Metall. No:95, January 1986, pp. A8-A14.

Bradley, S., Hax, A., Magnanti, T., 1977, Applied Mathematical Programming, Addison-Wesley Pub. Co., London.

Bray, J.M., Plummer, I.M., 1964, Methods of Ventilation Analysis, Mining Magazine, April 1964, pp. 224-237.

Bunday, B.D., 1984, Basic Optimization Methods, Edward Arnold Pub. Co.

Cross, H., 1936, Analysis of Flow in Networks of Conduits of Conductors, Bulletin of Illionis Un. Eng. Exp. Station, No:286.

Dano, S., 1975, Nonlinear and Dynamic Programming, Springer-Verlag 220 p.

Davidon, W.C., 1959, Variable Metric Method for Minimization, AEC Research and Dev. Report ANL-5990.

Ergün, İ., 1973, Kömür Ocaklarında Havalandırma Şebekeleri Hesaplarının Kompüterle Yapılması, Türkiye Madencilik Bilimsel ve Teknik 3. Kongresi, s. 473-493.

KAYNAKLAR DİZİNİ (Devam ediyor)

Fiacco, A., McCormick, G.P., 1968, Nonlinear Programming-Sequential Unconstrained Minimization Techniques, John Wiley and Sons, New York.

Fletcher, R., Powell, J.D., 1963, A Rapidly Convergent Method for Minimization , Computer Journal, 6, pp. 163-168.

Fletcher, R., Reeves C.M., 1964, Function Minimization by Conjugate Gradients, Computer Journal, 7, pp. 149-154.

Ford, L.R., Fulkerson, D.R., 1960, Flows in Networks, Princeton University Press.

Fox, R., 1973, Optimization Methods for Engineering Design, Addison-Wesley Pub. Co., 270 p.

Gaver, D., Thompson, G., 1973, Programming and Probability Models in Operations Research, Brooks-Cole Pub. Co.

Gerald, C., 1970, Applied Numerical Analysis, Addison-Wesley Pub. Co., 334 p.

Gottfried, B., Weisman, J., 1973, Introduction to Optimization Theory, Prentice-Hall Inc., Englewood Cliffs, N.J., 570 p.

Güney, M., 1973, E.K.İ. Üzülmez Bölgesi Havalandırma Sisteminin Etüdü, Türkiye Madencilik Bilimsel ve Teknik 3. Kongresi, s. 409-439.

Gürtan, K., 1982, İstatistik ve Araştırma Metodları, İ.Ü. Yayınları, No:2941, 831 s.

Güyagüler, T., 1979, Ocak Havalandırmasında Bilgisayar Programı, Türkiye Madencilik Bilimsel ve Teknik 6. Kongresi, No:20.

Güyagüler, T., 1981, Ocak Havalandırması-Yeraltı Kömür Madenciliğinde Çevre Sorunları ve Kontrol Yöntemleri Seminer El Kitabı.

Güyagüler, T., 1982, Değişik Birimlerde Havayolu Direnci ve Sürtünme Katsayıları, Madencilik Dergisi, Eylül 1982, s. 15-26.

Hadley, G., 1964, Nonlinear and Dynamic Programming, Addison-Wesley Co., 484 p.

Hall, C.J., 1981, Mine Ventilation Engineering, Society of Mining Engineers, New York.

Hall, A.E., Stoakes, M.A., Gangal, M.K., 1982, CANMET's Thermodynamic Ventilation Network Program, CIM Bull., December 1982, pp. 52-60.

KAYNAKLAR DİZİNİ (Devam ediyor)

Hartman, L.H., Trafton, B.O., 1963, Digital Computer May Find New Use in Determining Mine Ventilation Networks, Mining Engineers, September 1963, pp. 39-43.

Hartman, L.H., 1982, Mine Ventilation and Air Conditioning, 2nd Ed., John Wiley and Sons, New York, 790 p.

Hillier, F., Lieberman, G., 1974, Operations Research, Holden-Day Inc., 2nd Ed., 749 p.

Himmelblau, D.M., 1972, Applied Nonlinear Programming, McGraw-Hill Co., 498 p.

Jeffrey, A., 1979, Mathematics for Engineers and Scientists, 2nd Ed., Thomas Nelson Ltd.

Jensen, P.A., Barnes, J.W., 1980, Network Flow Programming, John Wiley and Sons, New York, 408 p.

Kara, İ., 1985, Yöneylem Araştırmasının Yöntembilimi, Anadolu Üniversitesi Yayınları, No:96, 117 s.

Kara, İ., 1986, Yöneylem Araştırması-Doğrusal Olmayan Modeller, Anadolu Üniversitesi Yayınları, No:139, 346 s.

Koo, D., 1977, Elements of Optimization, Springer-Verlag Co., 220 p.

Kowalik, J., Osborne, M.R., 1968, Methods for Unconstrained Optimization Problems, Elsevier Co., New York.

Kunzi, H.P., Krelle, W., 1966, Nonlinear Programming, Blaisdell Co.

Leon, A., 1966, A Comparison Among Eight Known Optimizing Procedures- Recent Advances in Optimization Techniques- Editor, Lavi, A., Vogl, T., John Wiley, pp. 23-47

Loomba, N., Turban, E., 1974, Applied Programming for Management, Holt, Rinehart, Winsten Co.

Luenberger, D., 1969, Optimization by Vector Space Methods, John Wiley and Sons, 283 p.

Luenberger, D., 1973, Introduction to Linear and Nonlinear Programming, Addison-Wesley Co., 356 p.

McCormick, G.P., 1983, Nonlinear Programming-Theory, Algorithms and Applications, John Wiley and Sons, 444 p.

KAYNAKLAR DİZİNİ (Devam ediyor)

McPherson, M.J., 1966, Ventilation Network Analysis By Digital Computers, The Mining Engineer, Trans. Vol. 126, No:73, pp. 13-27.

Murtagh, B.A., 1970, A Short Description of the Variable Metric Method - Integer and Nonlinear Programming - Editor : Abadie, J., North Holland Co., pp.525-529.

Ortega, J.M., Rheinboldt, W.C., 1970, Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, 572 p.

Pearson, J.D., 1969, Variable Metric Methods of Minimization, Computer Journal, 12, pp.171-178.

Phillips, D.T., Ravindran, A., Solber, G.J., 1976, Operations Research - Principles and Practice, John Wiley and Sons, 585 p.

Phillips, D.T., Garcia-Diaz, A., 1981, Fundamentals of Network Analysis, Prentice-Hall Inc., N.J.

Pierre, D., 1969, Optimization Theory With Application, John Wiley and Sons, 612 p.

Polak, E., 1971, Computational Methods in Optimization- A Unified Approach, Academic Press, New York.

Saltoğlu, S., 1975, Madenlerde Havalandırma ve Emniyet İşleri, İ.T.Ü. Kütüphanesi, No:1019, 313 s.

Schwefel, H.P., 1981, Numerical Optimization of Computer Models, John Wiley and Sons, 388 p.

Simmons, D., 1975, Nonlinear Programming for Operations Research, Prentice-Hall Inc.

Sivazlian, B.D., Stanfel, L.E., 1975, Optimization Techniques in Operations Research, Prentice-Hall Inc., 502 p.

Skochinsky, A., Komarov, V., 1969, Mine Ventilation, MIR Publisher, Moscow.

Şenel, M., 1983, Nümerik Analiz, Anadolu Üniversitesi, Eskişehir.

Taha, H., 1976, Operations Research-An Introduction, 2nd. Ed. McMillan Pub. Co., 531 p.

Topçu, A., 1987, Mikro Mühendis-Mikrobilgisayar Paket Programları 3, Yayına Hazır Eser, Anadolu Üniversitesi, Eskişehir.

Tulunay, Y., 1980, Matematik Programlama ve İşletme Uygulamaları, İ.Ü. Yayınları,No:2721, 612 s.

KAYNAKLAR DİZİNİ (Devam ediyor)

Ueng, Y., Wang, Y.J., 1984, Analysis of Mine Ventilation Networks Using Nonlinear Programming Techniques, Int. Journal of Min. Eng., No:2, pp.245-252.

Wagner, H., 1969, Principles of Operations Research, Prentice-Hall Inc., N.J.

Walsh, G.R., 1979, Methods of Optimization, John Wiley and Sons, New York, 220 p.

Wang, Y.J., 1982, Ventilation Network Theory -Mine Ventilation and Air Conditioning- Editor : Hartman, L.H., 2nd Ed., John Wiley and Sons, New York, pp.483-516.

Wang, Y.J., 1984, A Nonlinear Programming Formulation for Mine Ventilation Networks With Natural Splitting, Int. Journal Rock Mech. Min. Sci. Geomech. Abs., Vol:21, No:1, pp.43-45.

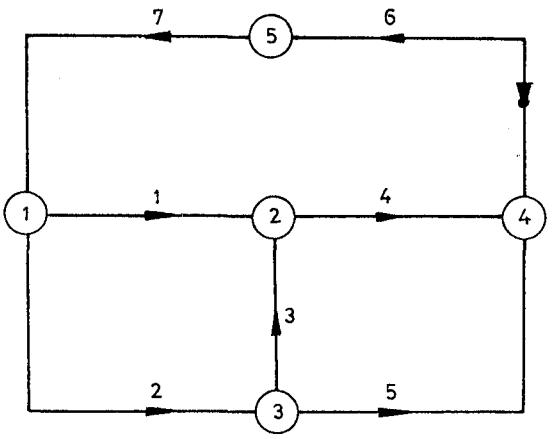
Whittle, P., 1971, Optimization Under Constraints, John Wiley and Sons, New York, 241 p.

Wilde, D.J., 1964, Optimum Seeking Methods, Prentice-Hall Inc., N.J.

Zahradnik, R.L., 1971, Theory and Techniques of Optimization for Practicing Engineers, Barnes-Nobel Inc., 326 p.

Zoutendijk, G., 1970, Nonlinear Programming Computational Methods, -Integer and Nonlinear Programming- Editor : Abadie, j., North Holland Co. pp. 37-87.

EK-1 : Örnek Şebeke-1



Kollar	1	2	3	4	5	6	7
Dirençler (Murg)	1000	400	80	600	900	2500	500

SEÇİLEN GÖZLER

GÖZ 1 : 6 7 2 3 4

GÖZ 2 : 5 -4 -3

GÖZ 3 : 1 -3 -2

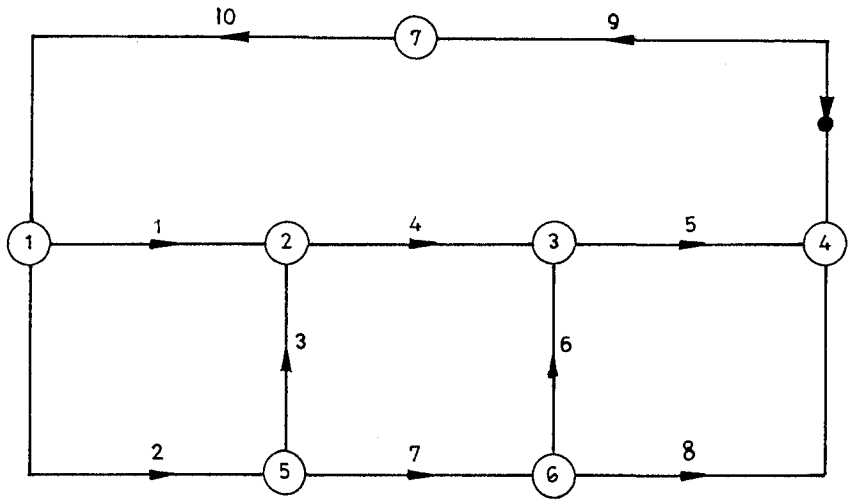
VANTİLATÖR KATSAYILARI

A = 494,54

B = -6,9

C = 0,047

EK-2 : Örnek Şebeke-2



Kollar	1	2	3	4	5	6	7	8	9	10
Dirençler (Murg)	4000	300	90	100	400	60	600	900	1500	500

SEÇİLEN GÖZLER

GÖZ 1 : 9 10 2 3 4 5

GÖZ 2 : 7 -6 -4 -3

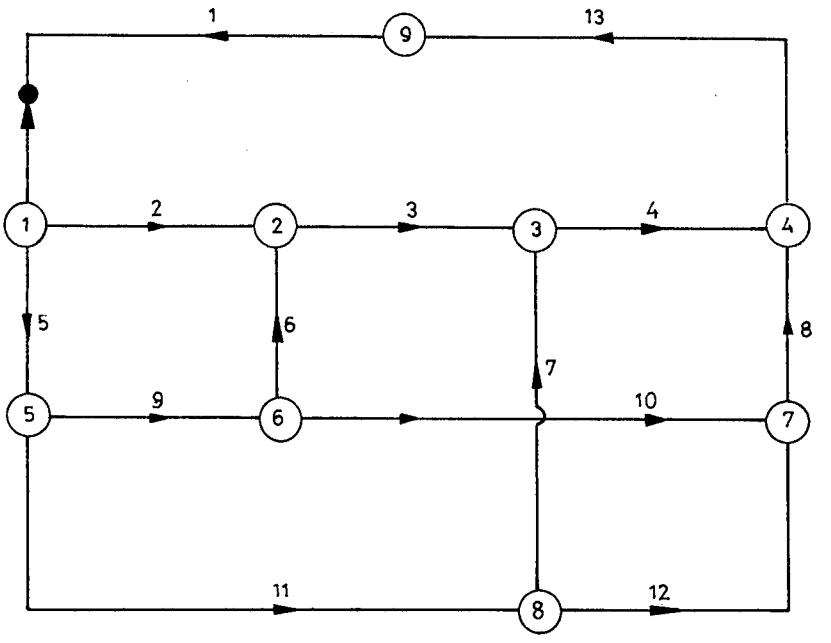
GÖZ 3 : 8 -5 6

GÖZ 4 : 1 -3 -2

VANTİLATÖR KATSAYILARI

A = 268,293 B = 0,77124 C = $-5,7144.10^{-2}$

EK-3 : Örnek Şebeke-3



Kol	1	2	3	4	5	6	7	8	9	10	11	12	13
Direnç (Murg)	2700	1000	600	800	300	400	300	300	900	2000	5000	900	300

SEÇİLEN GÖZLER

GÖZ 1 : 1 5 9 6 3 4 13

GÖZ 2 : 12 8 -4 -7

GÖZ 3 : 2 -6 -9 -5

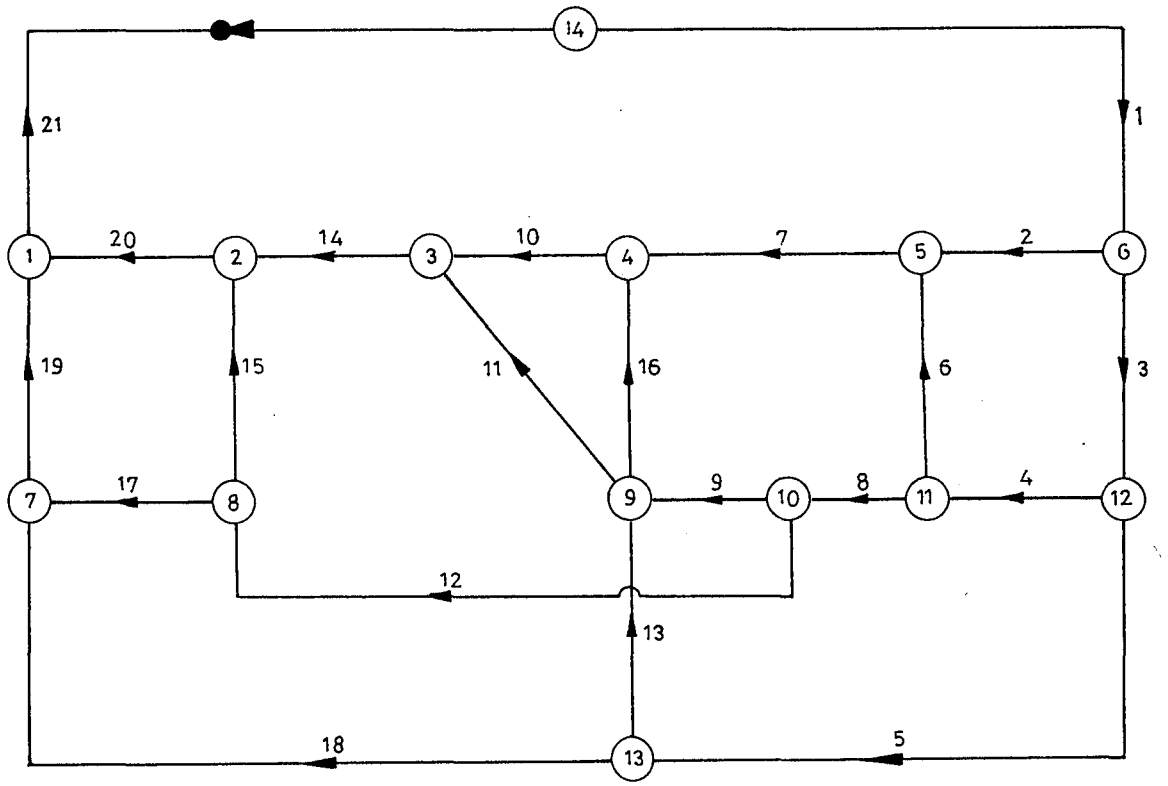
GÖZ 4 : 10 8 -4 -3 -6

GÖZ 5 : 11 7 -3 -6 -9

VANTİLATÖR KATSAYILARI

A = 341,227 B = -0,55572 C = $-5,7623.10^{-3}$

EK-4 : Örnek Şebeke-4



Kol	1	2	3	4	5	6	7	8	9	10	11
Direnç (Murg)	1,1	10000	0,5	14,8	21	252,7	47,6	7,9	6,6	13,6	215
Kol	12	13	14	15	16	17	18	19	20	21	
Direnç (Murg)	19,2	257,9	19,2	244	45,2	1000	3000	0,9	39,4	1,9	

SEÇİLEN GÖZLER

GÖZ 1 : 21 1 3 4 8 9 16 10 14 20
GÖZ 2 : 11 -10 -16
GÖZ 3 : 15 -14 -10 -16 -9 12
GÖZ 4 : 6 7 -16 -9 -8
GÖZ 5 : 13 -9 -8 -4 5
GÖZ 6 : 17 19 -20 -14 -10 -16 -9 12
GÖZ 7 : 18 19 -20 -14 -10 -16 -9 -8 -4 5
GÖZ 8 : 2 7 -16 -9 -8 -4 -3

VANTİLATÖR KATSAYILARI

A = 341,227 B = -0,55572 C = $-5,7623.10^{-3}$

EK-5 : Hardy Cross Yöntemi Bilgisayar Programı

```
10 OPEN "KR:" AS FILE 1
20 ! ********************************************
30 ! *            HARDY CROSS                   *
40 ! *       ITERASYON TEKNIGI ILE              *
50 ! *    HAVALANDIRMA SEBEKELERININ ANALIZI    *
60 ! *             SAIM SARAC : 10-11-1985      *
70 ! ********************************************
80 EXTEND
90 ; #1 CHR$(29%)
100 DIM T$=25%
110 T$="1986-6-12 00:00:00"
120 SET TIME T$
130 READ Kol,Goz,Vs
140 DIM Q(Kol),R(Kol),G(Goz),Gozk(Goz,Kol)
150 DIM A1(Vs),B1(Vs),C1(Vs),Vk(Vs)
160 FOR I=1 TO Vs
170   READ A1(I),B1(I),C1(I),Vk(I)
180 NEXT I
190 Has=.005
200 FOR I=1 TO Kol : READ R(I) : NEXT I
210 FOR I=1 TO Kol : READ Q(I) : NEXT I
220 FOR I=1 TO Goz
230   READ G(I)
240   FOR J=1 TO G(I)
250     READ Gozk(I,J)
260   NEXT J
270 NEXT I
280 It=0
290 Dmax=0
300 FOR K2=1 TO Goz
310   Pay=0 : Pyd=0 : K1=G(K2)
320   FOR J=1 TO K1
330     Ka=Gozk(K2,J)
340     IF Ka<0 THEN 370
350     C=1
360     GOTO 390
370     C=-1
380     Ka=-Ka
390     Q1=ABS(Q(Ka))
400     FOR L1=1 TO Vs
410       IF Ka=Vk(L1) GOTO 440
420     NEXT L1
430     Van=0 : Van1=0 : GOTO 460
440     Van=A1(L1)+B1(L1)*Q1+C1(L1)*Q1^2
450     Van1=B1(L1)+2*C1(L1)*Q1
460     Pay=Pay+C*(R(Ka)*Q(Ka)*Q1-Van)
470     Pyd=Pyd+(2*R(Ka)*Q1-Van1)
480   NEXT J
490   Dq=-Pay/Pyd
500   Dm=ABS(Dq)
510   IF (Dm-Dmax)<=0 THEN 530
520   Dmax=Dm
530   FOR J=1 TO K1
540     Ka=Gozk(K2,J)
550     IF Ka>=0 THEN 590
560     C=-1
```

```
570 Ka=-Ka
580 GOTO 600
590 C=1
600 Q(Ka)=Q(Ka)+C*Dq
610 NEXT J
620 NEXT K2
630 It=It+1
640 ; #1 It,Dmax
650 IF (Dmax-Has)<0 THEN 670
660 GOTO 290
670 ; #1 : ; #1
680 ; #1 TAB(2) "KOL" TAB(7) "DIRENC" TAB(18) "DEBI" TAB(29) "BAS.KAYBI"
690 ; #1 "---------------------------------------"
700 FOR I=1 TO Kol
710   Dpc=R(I)*Q(I)^2/1000
720   ; #1 TAB(2) I TAB(7) R(I)*1000 TAB(16) Q(I) TAB(28) Dpc
730 NEXT I
740 D$=MID$(TIME$,11,9)
750 ; #1
760 ; #1 'ITERASYON SAYISI=';It
770 ; #1 "HESAPLAMA SURESI=";D$
780 END
1000 DATA 7,3,1,494.54,-6.9,.047,6
1010 DATA 1,.4,.08,.6,.9,2.5,.5
1020 DATA 0,0,0,0,0,0,0
1030 DATA 5,6,7,2,3,4
1040 DATA 3,5,-4,-3
1050 DATA 3,1,-3,-2
```

EK-6 : En Hızlı İniş Yöntemi (Altın Oran Tekniğini Kullanan) Bilgisayar Programı.

```
10 OPEN "PR:" AS FILE 1
20 ; #1 CHR$(29)
30 ! ************************************
40 ! *        GRADYAN YONTEMI           *
50 ! *              ILE                 *
60 ! *       (ALTIN ORAN ARAMA          *
70 ! *      TEKNIGINI KULLANARAK)       *
80 ! *    HAVALANDIRMA SEBEKELERININ    *
90 ! *            ANALIZI               *
95 ! *       15.6.1986..SAIM SARAC      *
98 ! ************************************
120 EXTEND
130 DIM T$=25%
140 T$="1986-10-28 00:00:00:"
150 SET TIME T$
160 READ Kol,Goz,Vs
170 DIM G(Goz),Gozk(Goz,Kol),R(Kol),Q(Kol),A(Vs),B(Vs),C(Vs),Vk(Vs)
180 DIM P(Kol),Ak(Goz),Q1(Kol),Qr(Kol)
190 FOR I=1 TO Vs
200 READ A(I),B(I),C(I),Vk(I) : NEXT I
210 FOR I=1 TO Kol
220 READ R(I) : Q(I)=0 : NEXT I
230 FOR I=1 TO Goz
240   READ G(I)
250   FOR J=1 TO G(I)
260   READ Gozk(I,J) : NEXT J
270   Ak(I)=ABS(Gozk(I,1))
280 NEXT I
290 FOR J=1 TO Kol
300 P(J)=0 : NEXT J
310 FOR I=1 TO Goz : T=0
320   FOR J=1 TO G(I) : Van=0
330     V=ABS(Gozk(I,J))
340     FOR K=1 TO Vs
350       IF V=Vk(K) GOTO 370
360     NEXT K : GOTO 380
370     Van=A(K)+B(K)*Q(V)+C(K)*Q(V)^2
380     T=T+R(V)*Q(V)*ABS(Q(V))*SGN(Gozk(I,J))-Van
390   NEXT J
400   P(Ak(I))=-T
410   FOR J=2 TO G(I)
420     V=ABS(Gozk(I,J))
430     P(V)=P(V)+P(Ak(I))*SGN(Gozk(I,J))
440   NEXT J
450 NEXT I : S8=0
460 GOSUB 1020
470 FOR I=1 TO Goz : W=Q(Ak(I))
480   FOR J=1 TO G(I)
490     V=ABS(Gozk(I,J))
500     Q(V)=Q(V)+X*P(Ak(I))*SGN(Gozk(I,J))
510   NEXT J
520   IF ABS(Q(Ak(I))-W)<.005 THEN S8=S8+1
530 NEXT I : ! ; #1
540 It=It+1 : IF S8<Goz GOTO 290
550 ; #1 TAB(2) "KOL" TAB(12) "DEBILER"
560 ; #1 "-------------------"
```

EK-6 : (Devam ediyor)

```
570 FOR I=1 TO Kol
580 ; #1 TAB(2) I TAB(12) Q(I) : NEXT I : ; #1
590 D$=MID$(TIME$,11,9)
600 ; #1 : ; #1 It;" .ITERASYONDA SON" : ; #1
610 ; #1 "HESAPLAMA SURESI=";D$
620 END
1000 ! GOLDEN SECTION YONTEMIYLE ARAMA
1010 ! ******************************
1020 A1=0 : A2=.1 : R=.5*(SQR(5)-1) : H=A2-A1
1030 X1=A1+R^2*H : Xr=A1+R*H : E=0
1040 FOR I=1 TO Goz
1050   FOR J=1 TO G(I)
1060     V=ABS(Gozk(I,J))
1070     IF E=2 GOTO 1100
1080     Q1(V)=Q1(V)+(Q(Ak(I))+P(Ak(I))*X1)*SGN(Gozk(I,J))
1090     IF E=1 GOTO 1110
1100     Qr(V)=Qr(V)+(Q(Ak(I))+P(Ak(I))*Xr)*SGN(Gozk(I,J))
1110   NEXT J
1120 NEXT I
1130 T=0 : S=0
1140 FOR I=1 TO Kol : Van=0 : V1=0
1150   FOR K=1 TO Vs
1160     IF I=Vk(K) GOTO 1180
1170   NEXT K : GOTO 1200
1180   IF E()2 Van=A(K)*Q1(I)+B(K)/2*Q1(I)^2+C(K)/3*Q1(I)^3
1190   IF E()1 V1=A(K)*Qr(I)+B(K)/2*Qr(I)^2+C(K)/3*Qr(I)^3
1200   IF E=2 GOTO 1230
1210   T=T+R(I)/3*ABS(Q1(I)^3)-Van : Fx1=T
1220   IF E=1 GOTO 1240
1230   S=S+R(I)/3*ABS(Qr(I)^3)-V1 : Fxr=S
1240 NEXT I
1250 FOR I=1 TO Kol
1260 Q1(I)=0 : Qr(I)=0 : NEXT I
1270 IF Fx1>Fxr GOTO 1310
1280 A2=Xr : H=A2-A1 : Fxr=Fx1
1290 IF H(.005 GOTO 1340
1300 Xr=X1 : X1=A1+R^2*H : E=1 : GOTO 1040
1310 A1=X1 : H=A2-A1 : Fx1=Fxr
1320 IF H(.005 GOTO 1340
1330 X1=Xr : Xr=A1+R*H : E=2 : GOTO 1040
1340 X=(A1+A2)/2
1350 RETURN
2000 DATA 7,3,1,494.54,-6.9,.047,5
2010 DATA 1,.4,.00,.6,.9,2.5,.5
2020 DATA 5,6,7,2,3,4
2030 DATA 3,5,-4,-3
2040 DATA 3,1,-3,-2
```

EK-7 : En Hızlı İniş Yöntemi (Kareli İnterpolasyon Tekniğini Kullanan) Bilgisayar Programı.

```
10 OPEN "pr:" AS FILE 1
30 ! ************************************
40 ! *      EN HIZLI INIS YONTEMI       *
50 ! *              ILE                 *
60 ! *    HAVALANDIRMA SEBEKELERININ    *
70 ! *           ANALIZI                *
80 ! *       15.6.1986..SAIM SARAC      *
90 ! ************************************
100 EXTEND : DIM T$=25%
120 T$="1986-10-28 00:00:00:" : SET TIME T$
140 READ Kol,Goz,Vs
150 DIM G(Goz),Gk(Goz,Kol),R(Kol),Q(Kol),A(Vs),B(Vs),C(Vs),Vk(Vs)
160 DIM P(Kol),Ak(Goz),X9(4),F9(4)
162 FOR I=1 TO Vs
164   READ A(I),B(I),C(I),Vk(I)
168 NEXT I
170 FOR I=1 TO Kol
180 READ R(I) : Q(I)=0 : NEXT I
190 FOR I=1 TO Goz
200   READ G(I)
210   FOR J=1 TO G(I)
220   READ Gk(I,J) : NEXT J
230   Ak(I)=ABS(Gk(I,1))
240 NEXT I
250 FOR J=1 TO Kol : P(J)=0 : NEXT J
270 FOR I=1 TO Goz : T=0
280   FOR J=1 TO G(I) : Van=0
290     V=ABS(Gk(I,J))
300     FOR K=1 TO Vs
302       IF V=Vk(K) GOTO 310
304     NEXT K : GOTO 320
310     Van=A(K)+B(K)*Q(V)+C(K)*Q(V)^2
320     T=T+R(V)*Q(V)*ABS(Q(V))*SGN(Gk(I,J))-Van
330   NEXT J : P(Ak(I))=-T
340   P(Ak(I))=-T
350 NEXT I : S8=0
360 FOR I=1 TO Goz
370   FOR J=2 TO G(I)
380     V=ABS(Gk(I,J))
390     P(V)=P(V)+P(Ak(I))*SGN(Gk(I,J))
400   NEXT J
410 NEXT I
420 GOSUB 2000
430 FOR I=1 TO Goz : W=Q(Ak(I))
440   FOR J=1 TO G(I)
450     V=ABS(Gk(I,J))
460     Q(V)=Q(V)+X*P(Ak(I))*SGN(Gk(I,J))
470   NEXT J
480   IF ABS(Q(Ak(I))-W)<.005 THEN S8=S8+1
490 NEXT I
510 It=It+1 : IF S8<Goz GOTO 250
520 ; #1 TAB(2) "KOL" TAB(12) "DEBILER"
530 ; #1 "--------------------"
```

EK-7 : (Devam ediyor)

```
540 FOR I=1 TO Kol
550 ; #1 TAB(2) I TAB(12) Q(I) : NEXT I : ; #1
560 D$=MID$(TIME$,11,9)
570 ; #1 : ; #1 It;" .ITERASYONDA SON" : ; #1
580 ; #1 "HESAPLAMA SURESI=";D$
590 END
2000 REM Optimal adim boyutunun kareli interpolasyon ile hesaplanmasi
2010 REM ************************************************************
2020 A9=.05 : H9=.025 : E9=.001 : T7=H9
2030 X9(1)=A9 : X9=X9(1) : GOSUB 2500 : F9(1)=Z9
2040 X9(2)=A9+H9 : X9=X9(2) : GOSUB 2500 : F9(2)=Z9
2050 IF F9(1))F9(2) GOTO 2100
2060 X9(3)=A9-H9 : X9=X9(3) : GOSUB 2500 : F9(3)=Z9
2070 IF F9(3))F9(1) GOTO 2140
2080 H9=H9+H9/4 : F9(2)=F9(1) : X9(2)=X9(1)
2090 F9(1)=F9(3) : X9(1)=X9(3) : GOTO 2060
2100 X9(3)=A9+2*H9 : X9=X9(3) : GOSUB 2500 : F9(3)=Z9
2110 IF F9(3))F9(2) GOTO 2140
2120 H9=H9*2 : F9(1)=F9(2) : X9(1)=X9(2)
2130 F9(2)=F9(3) : X9(2)=X9(3) : GOTO 2100
2140 Dn=(X9(2)-X9(3))*F9(1)
2150 Dn=Dn+(X9(3)-X9(1))*F9(2)+(X9(1)-X9(2))*F9(3)
2160 Nm=(X9(2)*X9(2)-X9(3)*X9(3))*F9(1)
2170 Nm=Nm+(X9(3)*X9(3)-X9(1)*X9(1))*F9(2)
2180 Nm=Nm+(X9(1)*X9(1)-X9(2)*X9(2))*F9(3)
2190 X9(4)=Nm/(2*Dn) : X9=X9(4) : GOSUB 2500 : F9(4)=Z9
2200 FOR J%=1 TO 3
2210   FOR K%=J%+1 TO 4
2220     IF F9(J%)(=F9(K%) GOTO 2250
2230     X9=X9(J%) : X9(J%)=X9(K%) : X9(K%)=X9
2240     F9=F9(J%) : F9(J%)=F9(K%) : F9(K%)=F9
2250 NEXT K% : NEXT J%
2260 IF ABS(X9(1)-X9(2))(E9 GOTO 2390
2270 S1=SGN(X9(2)-X9(1)) : S2=SGN(X9(3)-X9(1))
2280 S3=SGN(X9(4)-X9(1))
2290 IF S1=S2 AND S1=-S3 THEN X9(3)=X9(4) : F9(3)=F9(4)
2300 Dn=(X9(2)-X9(3))*F9(1)+(X9(3)-X9(1))*F9(2)+(X9(1)-X9(2))*F9(3)
2310 IF ABS(Dn)(1.E-15 GOTO 520
2320 F9=(F9(1)-F9(2))/(2*Dn)
2330 F9=F9*(X9(2)-X9(3))*(X9(3)-X9(1))
2340 X9(4)=(X9(1)+X9(2))/2+F9
2350 X9=X9(4) : GOSUB 2500 : F9(4)=Z9
2360 IF X7=X9(4) X=X7 : GOTO 2400
2370 X7=X9(4)
2380 GOTO 2200
2390 X=X9(1)
2410 RETURN
```

EK-7 : (Devam ediyor)

```
2500 REM Fonksiyon degerini hesaplama
2502 REM ----------------------------
2510 Z9=0 : FOR I%=1 TO Kol : V9=0
2520   FOR K%=1 TO Vs
2522     IF I%=Vk(K%) GOTO 2530
2524   NEXT K% : GOTO 2550
2530   K8=(Q(I%)+P(I%)*X9)
2540   V9=A(K%)*K8+B(K%)/2*K8*K8+C(K%)/3*K8^3
2550   Z9=Z9+(R(I%)/3*(ABS(Q(I%)+P(I%)*X9))^3-V9)
2560 NEXT I%
2570 RETURN
3000 DATA 7,3,1
3010 DATA 494.54,-6.9,.047,5
3020 DATA 1,.4,.08,.6,.9,2.5,.5
3030 DATA 5,6,7,2,3,4
3040 DATA 3,5,-4,-3
3050 DATA 3,1,-3,-2
```

EK-8 : Fletcher-Reeves Yöntemi Bilgisayar Programı.

```
10 OPEN 'CON:' AS FILE 1 : ; #1 CHR$(29)
30 ! ************************************
40 ! *  FLETCHER-REEVES   YONTEMI      *
50 ! *              ILE               *
60 ! *   HAVALANDIRMA SEBEKELERININ    *
70 ! *           ANALIZI               *
80 ! *    3.10.1986 SAIM SARAC         *
90 ! ************************************
100 EXTEND : DIM T$=25%
120 T$="1986-11-5 00:00:00" : SET TIME D$
140 READ Kol,Goz,Vs
150 DIM Dlt(Kol),G(Goz),Gk(Goz,Kol),P2(Kol),X9(4),F9(4)
160 DIM R(Kol),Q(Kol),P(Kol),P1(Kol),Ak(Goz),A(Vs),B(Vs),C(Vs),Vk(Vs)
170 FOR I=1 TO Vs
180   READ A(I),B(I),C(I),Vk(I)
190 NEXT I
200 FOR I=1 TO Kol
210 READ R(I) : Q(I)=0 : R(I)=R(I)/1000 : NEXT I
220 FOR I=1 TO Goz
230   READ G(I)
240   FOR J=1 TO G(I)
250   READ Gk(I,J) : NEXT J
260   Ak(I)=Gk(I,1)
270 NEXT I : It=0
280 GOSUB 1000
290 GOSUB 3000
300 Ha=0
310 FOR I=1 TO Goz : W=Q(Ak(I))
320   FOR J=1 TO G(I)
330     V=ABS(Gk(I,J))
340     Q(V)=Q(V)+X*P(Ak(I))*SGN(Gk(I,J))
350   NEXT J
360   IF ABS(Q(Ak(I))-W)<.005 THEN Ha=Ha+1
365 NEXT I
366 ! FOR I=1 TO Kol
367 ! ; Q(I); : NEXT I : ;
390 Pyd=0 : It=It+1 : Z=Z+1 : IF Ha<>Goz GOTO 590
400 IF Z=Goz Z=0 : GOTO 280
410 FOR I=1 TO Goz : V=Ak(I)
430 Pyd=Pyd+Dlt(V)^2 : P2(V)=P(V) : NEXT I
440 GOSUB 1000
450 Pay=0
455 FOR I=1 TO Goz
460   Pay=Pay+Dlt(Ak(I))^2
470 NEXT I : Beta=Pay/Pyd
480 FOR I=1 TO Kol
490 P1(I)=0 : NEXT I
500 FOR I=1 TO Goz : W=Ak(I)
510   P(W)=P(W)+Beta*P2(W)
520   FOR J=2 TO G(I)
530     V=ABS(Gk(I,J))
540     P1(V)=P1(V)+P(W)*SGN(Gk(I,J)) : P(V)=P1(V)
550   NEXT J
560 NEXT I
```

EK-8 : (Devam ediyor).

```
570 GOSUB 3000
580 GOTO 300
590 ; #1 TAB(2) "KOL" TAB(13) "DEBILER"
600 ; #1 "----------------------"
610 FOR I=1 TO Kol
620   ; #1 TAB(2) I TAB(12) Q(I)
625 NEXT I : ; #1
630 ; #1 I%;" .ITERASYONDA SON" : ; #1
640 D$=MID$(TIME$,11,9)
650 ; #1 "HESAPLAMA SURESI=";D$
660 END
1000 REM GRADYAN VEKTORUNUN HESABI
1010 REM ************************
1012 FOR I=1 TO Kol
1014 P1(I)=0 : NEXT I
1020 FOR I=1 TO Goz : T=0
1030   FOR J=1 TO G(I) : Van=0
1040     V=ABS(Gk(I,J))
1050     FOR K1=1 TO Vs
1060       IF V=Vk(K1) GOTO 1080
1070     NEXT K1 : GOTO 1090
1080     Van=A(K1)+B(K1)*Q(V)+C(K1)*Q(V)^2
1090     T=T+R(V)*Q(V)*ABS(Q(V))*SGN(Gk(I,J))-Van
1100   NEXT J
1110   W=Ak(I) : D(W)=T : P(W)=-T
1120   FOR J=2 TO G(I)
1130     V=ABS(Gk(I,J))
1140     P1(V)=P1(V)+P(Ak(I))*SGN(Gk(I,J))
1150     P(V)=P1(V)
1160   NEXT J
1170 NEXT I
1180 RETURN
3000 REM OPTIMAL ADIMIN KARELI INTERPOLASYONLA HESAPLANMASI
3010 REM ***************************************************
3020 A9=.05 : H9=.025 : E9=.005
3030 X9(1)=A9 : X9=X9(1) : GOSUB 3500 : F9(1)=Z9
3040 X9(2)=A9+H9 : X9=X9(2) : GOSUB 3500 : F9(2)=Z9
3050 IF F9(1))F9(2) GOTO 3100
3060 X9(3)=A9-H9 : X9=X9(3) : GOSUB 3500 : F9(3)=Z9
3070 IF F9(3))F9(1) GOTO 3140
3080 H9=H9+H9/4 : F9(2)=F9(1) : X9(2)=X9(1)
3090 F9(1)=F9(3) : X9(1)=X9(3) : GOTO 3060
3100 X9(3)=A9+2*H9 : X9=X9(3) : GOSUB 3500 : F9(3)=Z9
3110 IF F9(3))F9(2) GOTO 3140
3120 H9=H9*2 : F9(1)=F9(2) : X9(1)=X9(2)
3130 F9(2)=F9(3) : X9(2)=X9(3) : GOTO 3100
3140 Dn=(X9(2)-X9(3))*F9(1)
3150 Dn=Dn+(X9(3)-X9(1))*F9(2)+(X9(1)-X9(2))*F9(3)
3160 Nn=(X9(2)*X9(2)-X9(3)*X9(3))*F9(1)
3170 Nn=Nn+(X9(3)*X9(3)-X9(1)*X9(1))*F9(2)
3180 Nn=Nn+(X9(1)*X9(1)-X9(2)*X9(2))*F9(3)
3190 X9(4)=Nn/(2*Dn) : X9=X9(4) : GOSUB 3500 : F9(4)=Z9
3200 FOR J%=1 TO 3
3210   FOR K%=J%+1 TO 4
3220     IF F9(J%)(=F9(K%) GOTO 3250
```

EK-8 : (Devam ediyor)

```
3230 X9=X9(J%) : X9(J%)=X9(K%) : X9(K%)=X9
3240 F9=F9(J%) : F9(J%)=F9(K%) : F9(K%)=F9
3250 NEXT K% : NEXT J%
3260 IF ABS(X9(1)-X9(2))(E9 GOTO 3390
3270 S1=SGN(X9(2)-X9(1)) : S2=SGN(X9(3)-X9(1))
3280 S3=SGN(X9(4)-X9(1))
3290 IF S1=S2 AND S1=-S3 X9(3)=X9(4) : F9(3)=F9(4)
3300 Dn=(X9(2)-X9(3))*F9(1)+(X9(3)-X9(1))*F9(2)+(X9(1)-X9(2))*F9(3)
3310 IF ABS(Dn)(1.E-15 GOTO 590
3320 F9=(F9(1)-F9(2))/2/Dn
3330 F9=F9*(X9(2)-X9(3))*(X9(3)-X9(1))
3340 X9(4)=(X9(1)+X9(2))/2+F9
3350 X9=X9(4) : GOSUB 3500 : F9(4)=Z9
3360 IF X7=X9(4) X=X7 : GOTO 3400
3370 X7=X9(4)
3380 GOTO 3200
3390 X=X9(1)
3410 RETURN
3500 REM Fonksiyon degerinin hesabi
3510 Z9=0 : FOR I%=1 TO Kol : V9=0
3520   FOR K%=1 TO Vs
3522     IF I%=Vk(K%) GOTO 3530
3524   NEXT K% : GOTO 3550
3530   K8=Q(I%)+P(I%)*X9
3540   V9=A(K%)*K8+B(K%)/2*K8*K8+C(K%)/3*K8^3
3550   Z9=Z9+(R(I%)/3*(ABS(Q(I%)+P(I%)*X9))^3-V9)
3560 NEXT I% : RETURN
4000 DATA 7,3,1,494.54,-6.9,.047,6
4010 DATA 1,.4,.08,.6,.9,2.5,.5
4020 DATA 5,6,7,2,3,4
4030 DATA 3,5,-4,-3
4040 DATA 3,1,-3,-2
```

EK-9 : Newton Yöntemi Bilgisayar Programı.

```
10 OPEN "pr:" AS FILE 1
20 ; #1 CHR$(29) : EXTEND
30 ! ************************************
40 ! *       NEWTON YONTEMI ILE         *
50 ! *   HAVALANDIRMA SEBEKELERININ     *
60 ! *        ANALIZI PROGRAMI          *
70 ! *   SAIM SARAC          5/2/1987   *
80 ! ************************************
100 READ Kol,Goz,Vs
110 DIM Goz(Goz),Gozk(Goz,Kol),H(Goz,Goz),B(Goz,Goz),Ak(Goz),D(Goz)
120 DIM R(Kol),Q(Kol),A1(Vs),B1(Vs),C1(Vs),Vk(Vs)
130 DIM T$=25% : I%=0
140 T$="1987-2-10 00:00:00" : SET TIME T$
160 FOR I=1 TO Vs
170 READ A1(I),B1(I),C1(I),Vk(I) : NEXT I
180 FOR I=1 TO Kol : READ R(I) : NEXT I
190 FOR I=1 TO Kol : READ Q(I) : NEXT I
200 FOR I=1 TO Goz
210   READ Goz(I)
220   FOR J=1 TO Goz(I)
230     READ Gozk(I,J)
240   NEXT J : Ak(I)=ABS(Gozk(I,1))
250 NEXT I
260 REM HESSIEN MATRISININ OLUSTURULMASI
280 FOR I=1 TO Goz
290   FOR J=1 TO Goz
300     IF I=J THEN B(I,J)=1 ELSE B(I,J)=0
310     H(I,J)=0
320 NEXT J : NEXT I
330 FOR I=1 TO Goz : T=0
340   FOR J=1 TO Vs
350     IF Ak(I)=Vk(J) GOTO 380
360   NEXT J
370   Van=0 : GOTO 390
380   Van=B1(J)+2*C1(J)*Q(Ak(I))
390   FOR K=1 TO Goz(I)
400     L=ABS(Gozk(I,K))
410     T=T+2*R(L)*Q(L)
420   NEXT K
430 H(I,I)=T-Van : NEXT I
440 FOR I=1 TO (Goz-1)
450   FOR J=2 TO Goz(I)
460     R=Gozk(I,J)
470     FOR K=I+1 TO Goz
480       FOR L=2 TO Goz(K)
490         Z=ABS(Gozk(K,L))
500         IF ABS(R)<>Z GOTO 530
510         H(I,K)=H(I,K)+SGN(R)*SGN(Gozk(K,L))*2*R(Z)*Q(Z) : GOTO 540
530       NEXT L
540     NEXT K
550   NEXT J
560 NEXT I
```

EK-9 : (Devam ediyor)

```
570 FOR I=2 TO Goz
580   FOR J=1 TO I-1
590     H(I,J)=H(J,I)
600   NEXT J
610 NEXT I
620 REM HESSIEN MATRISININ INVERSI
640 A$="DUZENLI" : G9=0
650 FOR I9=1 TO Goz
660   T9=0 : FOR J9=1 TO Goz
670     T9=T9+ABS(H(I9,J9))
680   NEXT J9
690   IF T9>G9 THEN G9=T9
700 NEXT I9
710 G9=G9*.000001
720 REM PIVOT ARAMA
730 FOR I9=1 TO Goz-1
740   T9=0 : V9=0
750   FOR J9=I9 TO Goz
760     IF ABS(H(J9,I9))>T9 THEN T9=ABS(H(J9,I9)) : V9=J9
770   NEXT J9
780   IF T9<G9 GOTO 1130
790   IF I9=V9 GOTO 870
800   FOR J9=I9 TO Goz
810     A9=H(I9,J9) : H(I9,J9)=H(V9,J9) : H(V9,J9)=A9
820   NEXT J9
830   FOR J9=1 TO Goz
840     A9=B(I9,J9) : B(I9,J9)=B(V9,J9) : B(V9,J9)=A9
850   NEXT J9
860   REM INDIRGEME
870   T9=H(I9,I9)
880   FOR J9=I9+1 TO Goz
890     A9=H(J9,I9)/T9
900     FOR K9=I9+1 TO Goz
910       H(J9,K9)=H(J9,K9)-A9*H(I9,K9)
920     NEXT K9
930     FOR K9=1 TO Goz
940       B(J9,K9)=B(J9,K9)-A9*B(I9,K9)
950     NEXT K9
960   NEXT J9
970 NEXT I9
980 REM GERI HESAP
990 T9=H(Goz,Goz)
1000 FOR J9=1 TO Goz
1010   B(Goz,J9)=B(Goz,J9)/T9
1020 NEXT J9
1030 FOR I9=Goz-1 TO 1 STEP -1
1040   FOR J9=1 TO Goz
1050     A9=B(I9,J9)
1060     FOR K9=I9+1 TO Goz
1070       A9=A9-H(I9,K9)*B(K9,J9)
1080     NEXT K9
1090     B(I9,J9)=A9/H(I9,I9)
1100   NEXT J9
1110 NEXT I9 : GOTO 1140
```

EK-9 : (Devam ediyor)

```
1130 A$="TEKIL" : ; A$ : STOP
1140 GOSUB 2000
1150 S=0
1160 FOR I=1 TO Goz : T=0
1170   W=Q(Ak(I))
1180   FOR K=1 TO Goz
1190     T=T+B(I,K)*D(K)
1200   NEXT K
1210   Q(Ak(I))=Q(Ak(I))-T
1220   IF ABS(W-Q(Ak(I)))<.005 S=S+1
1230   FOR J=2 TO Goz(I)
1240     V=ABS(Gozk(I,J))
1250   Q(V)=0 : NEXT J
1260 NEXT I : It=It+1
1270 FOR I=1 TO Goz
1280   FOR J=2 TO Goz(I)
1290     V=ABS(Gozk(I,J))
1300     Q(V)=Q(V)+Q(Ak(I))*SGN(Gozk(I,J))
1310 NEXT J : NEXT I
1320 IF S<Goz GOTO 260
1330 ; #1 TAB(2) "KOL" TAB(8) "HAVA MIK."
1340 ; #1 STRING$(16,95)
1350 FOR I=1 TO Kol
1360 ; #1 TAB(2) I TAB(8) Q(I) : NEXT I : ; #1
1370 D$=MID$(TIME$,11,9)
1380 ; #1 "HESAPLAMA SURESI=";D$
1390 ; #1 "ITERASYON SAYISI=";It
1400 END
2000 REM GRADYAN VEKTORUNUN OLUSTURULMASI
2010 REM *********************************
2020 FOR I=1 TO Goz : T=0
2030   FOR J=1 TO Goz(I)
2040     V=ABS(Gozk(I,J))
2050     FOR K=1 TO Vs
2060       IF V=Vk(K) GOTO 2090
2070     NEXT K
2080     Van=0 : GOTO 2100
2090     Van=A1(K)+B1(K)*Q(V)+C1(K)*Q(V)^2
2100     T=T+R(V)*Q(V)*ABS(Q(V))*SGN(Gozk(I,J))-Van
2110   NEXT J
2120   D(I)=T
2130 NEXT I
2140 RETURN
2150 STOP
3000 DATA 7,3,1,494.54,-5.9,.047,6
3010 DATA 1,.4,.08,.6,.9,2.5,.5
3020 DATA 1,0,0,1,0,1,1
3030 DATA 5,6,7,2,3,4
3040 DATA 3,5,-4,-3
3050 DATA 3,1,-3,-2
```

EK-10 : Davidon-Fletcher-Powell Yöntemi Bilgisayar Programı.

```
10 OPEN 'CON:' AS FILE 1
30 ! ************************************
40 ! *    DAVIDON-FLETCHER-POWELL       *
50 ! *         YONTEMI  ILE             *
60 ! *   HAVALANDIRMA SEBEKELERININ     *
70 ! *           ANALIZI                *
80 ! *   12.11.1985 SAIM SARAC          *
90 ! ************************************
100 EXTEND : DIM T$=25%
120 T$="1986-11-5 00:00:00" : SET TIME T$
140 READ Kol,Goz,Vs
150 DIM Dlt(Kol),G(Goz),Gk(Goz,Kol),Pl(Kol),A(Vs),B(Vs),C(Vs),Vk(Vs)
160 DIM R(Kol),Q(Kol),P(Kol),Ak(Goz),Is(Kol),H(Goz,Goz)
170 DIM Pay(Goz,Goz),Z(Goz),Bl(Goz,Goz),X9(4),F9(4)
180 FOR I=1 TO Vs
190   READ A(I),B(I),C(I),Vk(I)
195 NEXT I
200 FOR I=1 TO Kol
210   READ R(I) : R(I)=R(I)/1000 : Q(I)=0
215 NEXT I
220 FOR I=1 TO Goz
230   READ G(I)
240   FOR J=1 TO G(I)
250   READ Gk(I,J) : NEXT J
260   Ak(I)=Gk(I,1)
270 NEXT I : It=0
280 FOR I=1 TO Goz
290   FOR J=1 TO Goz
300     IF I=J H(I,J)=1 : GOTO 320
310     H(I,J)=0
320   NEXT J
330 NEXT I
340 GOSUB 1000
350 FOR I=1 TO Kol
360 Pl(I)=0 : NEXT I
370 FOR I=1 TO Goz : T=0
380   FOR J=1 TO Goz
390     W1=ABS(Ak(J))
400     T=T-H(I,J)*Dlt(W1)
410   NEXT J
412   W=ABS(Ak(I))
420   P(W)=T
430   FOR J=2 TO G(I)
440     V=ABS(Gk(I,J))
450     Pl(V)=Pl(V)+P(W)*SGN(Gk(I,J))
460     P(V)=Pl(V)
470 NEXT J : NEXT I
480 GOSUB 3000
490 Ha=0
500 FOR I=1 TO Goz : W=ABS(Ak(I)) : I=G(W)
510   FOR J=1 TO G(I)
520     V=ABS(Gk(I,J))
530     Q(V)=Q(V)+X*P(W)*SGN(Gk(I,J))
540   NEXT J
```

EK-10 : (Devam ediyor)

```
550 IF ABS(Q(W)-Z)(.005 Ha=Ha+1
560 P1(W)=D(W)
570 NEXT I
590 It=It+1 : IF Ha)=Goz GOTO 830
600 GOSUB 1000
610 T=0
620 FOR I=1 TO Goz
630   W=ABS(Ak(I))
640   P1(W)=D(W)-P1(W)
642 NEXT I
644 FOR I=1 TO Goz : S=0
646   W=ABS(Ak(I))
650   FOR J=1 TO Goz
655     K3=ABS(Ak(J))
660     Pay(I,J)=X*P(W)*P(K3)
670     S=S+H(I,J)*P1(K3)
680   NEXT J
690   T=T+P(W)*P1(W) : Z(I)=S
700 NEXT I : S=0
710 FOR I=1 TO Goz
720   FOR J=1 TO Goz
730     B1(I,J)=Pay(I,J)/T
740     Pay(I,J)=-Z(I)*Z(J)
750   NEXT J
760   S=S+P1(ABS(Ak(I)))*Z(I)
770 NEXT I
780 FOR I=1 TO Goz
790   FOR J=1 TO Goz
800     H(I,J)=H(I,J)+B1(I,J)+Pay(I,J)/S
810 NEXT J : NEXT I
820 GOTO 350
830 ; #1 TAB(2) "KOL" TAB(12) "DEBILER"
840 ; #1 "--------------------"
850 FOR I=1 TO Kol
860 ; #1 TAB(2) I TAB(12) Q(I) : NEXT I : ; #1
870 D$=MID$(TIME$,11,9)
880 ; #1 It;" . ITERASYONDA SON" : ; #1
890 ; #1 "HESAPLAMA SURESI=";D$ : ; #1
900 ; #1 "KULLANILAN BAYT=";SYS(3)
910 END
1000 REM GRADYAN VEKTORUNUN HESABI
1010 REM ************************
1020 FOR I=1 TO Goz : T=0
1030   FOR J=1 TO G(I) : Van=0
1040     V=ABS(Gk(I,J))
1050     FOR K1=1 TO Vs
1060       IF V=Vk(K1) GOTO 1080
1070     NEXT K1 : GOTO 1090
1080     Van=A(K1)+B(K1)*Q(V)+C(K1)*Q(V)^2
1090     T=T+R(V)*Q(V)*ABS(Q(V))*SGN(Gk(I,J))-Van
1100   NEXT J
1110   W=ABS(Ak(I)) : D(W)=T
1120 NEXT I
1130 RETURN
```

EK-10 : (Devam ediyor)

```
3000 REM OPTIMAL ADIMIN KARELI INTERPOLASYONLA HESAPLANMASI
3010 REM ***************************************************
3020 A9=.1 : H9=.025 : E9=.005 : T7=H9
3030 X9(1)=A9 : X9=X9(1) : GOSUB 3500 : F9(1)=Z9
3040 X9(2)=A9+H9 : X9=X9(2) : GOSUB 3500 : F9(2)=Z9
3050 IF F9(1))F9(2) GOTO 3100
3060 X9(3)=A9-H9 : X9=X9(3) : GOSUB 3500 : F9(3)=Z9
3070 IF F9(3))F9(1) GOTO 3140
3080 H9=H9+H9/4 : F9(2)=F9(1) : X9(2)=X9(1)
3090 F9(1)=F9(3) : X9(1)=X9(3) : GOTO 3060
3100 X9(3)=A9+2*H9 : X9=X9(3) : GOSUB 3500 : F9(3)=Z9
3110 IF F9(3))F9(2) GOTO 3140
3120 H9=H9*2 : F9(1)=F9(2) : X9(1)=X9(2)
3130 F9(2)=F9(3) : X9(2)=X9(3) : GOTO 3100
3140 Dn=(X9(2)-X9(3))*F9(1)
3150 Dn=Dn+(X9(3)-X9(1))*F9(2)+(X9(1)-X9(2))*F9(3)
3155 IF ABS(Dn)(1.E-33 GOTO 830
3160 Nm=(X9(2)*X9(2)-X9(3)*X9(3))*F9(1)
3170 Nm=Nm+(X9(3)*X9(3)-X9(1)*X9(1))*F9(2)
3180 Nm=Nm+(X9(1)*X9(1)-X9(2)*X9(2))*F9(3)
3190 X9(4)=Nm/(2*Dn) : X9=X9(4) : GOSUB 3500 : F9(4)=Z9
3200 FOR J=1 TO 3
3210   FOR K=J+1 TO 4
3220     IF F9(J)(=F9(K) GOTO 3250
3230     X9=X9(J) : X9(J)=X9(K) : X9(K)=X9
3240     F9=F9(J) : F9(J)=F9(K) : F9(K)=F9
3250 NEXT K : NEXT J
3260 IF ABS(X9(1)-X9(2))(E9 GOTO 3390
3270 S1=SGN(X9(2)-X9(1)) : S2=SGN(X9(3)-X9(1))
3280 S3=SGN(X9(4)-X9(1))
3290 IF S1=S2 AND S1=-S3 X9(3)=X9(4) : F9(3)=F9(4)
3300 Dn=(X9(2)-X9(3))*F9(1)+(X9(3)-X9(1))*F9(2)+(X9(1)-X9(2))*F9(3)
3310 IF ABS(Dn)(1.E-33 X=X9(2) : GOTO 3410
3320 F9=(F9(1)-F9(2))/2/Dn
3330 F9=F9*(X9(2)-X9(3))*(X9(3)-X9(1))
3340 X9(4)=(X9(1)+X9(2))/2+F9
3350 X9=X9(4) : GOSUB 3500 : F9(4)=Z9
3360 IF X7=X9(4) X=X7 : GOTO 3410
3370 X7=X9(4)
3380 GOTO 3200
3390 X=X9(1)
3410 RETURN
3500 REM Fonksiyon degerinin hesabi
3510 Z9=0 : FOR I%=1 TO Kol : V9=0
3520   FOR K%=1 TO Vs
3522     IF I%=Vk(K%) GOTO 3530
3524   NEXT K% : GOTO 3550
3530   K8=Q(I%)+P(I%)*X9
3540   V9=A(K%)*K8+B(K%)/2*K8*K8+C(K%)/3*K8^3
3550   Z9=Z9+(R(I%)/3*(ABS(Q(I%)+P(I%)*X9))^3-V9)
3560 NEXT I% : RETURN
4000 DATA 7,3,1,494.54,-6.9,.047,6
4010 DATA 1,.4,.08,.6,.9,2.5,.2
4020 DATA 5,6,7,2,3,4
4030 DATA 3,5,-4,-3
4040 DATA 3,1,-3,-2
```

EK-11 : Kombine Yöntem Bilgisayar Programı.

```
10 OPEN "PR:" AS FILE 1
20 ; #1 CHR$(29)
30 ! ************************************
40 ! *        KOMBINE YONTEM            *
50 ! *             ILE                  *
60 ! *    HAVALANDIRMA SEBEKELERININ    *
70 ! *           ANALIZI                *
80 ! *       15.6.1986..SAIM SARAC      *
90 ! ************************************
100 EXTEND
110 DIM T$=25%
120 T$="1986-10-28 00:00:00:"
130 SET TIME T$
140 READ Kol,Goz,Vs
150 DIM Goz(Goz),Gozk(Goz,Kol),R(Kol),Q(Kol)
160 DIM P(Kol),Ak(Goz),A(Vs),B(Vs),C(Vs),Vk(Vs),X9(4),F9(4)
170 FOR I=1 TO Vs
180   READ A(I),B(I),C(I),Vk(I)
190 NEXT I
200 FOR I=1 TO Kol
210 READ R(I) : Q(I)=0 : R(I)=R(I)/1000 : NEXT I
220 FOR I=1 TO Goz
230   READ Goz(I)
240   FOR J=1 TO Goz(I)
250   READ Gozk(I,J) : NEXT J
260   Ak(I)=Gozk(I,1)
270 NEXT I : It=0
280 FOR J=1 TO Kol
290 P(J)=0 : NEXT J
300 FOR I=1 TO Goz : T=0
310   FOR J=1 TO Goz(I) : Van=0
320     V=ABS(Gozk(I,J))
330     FOR K3=1 TO Vs
340       IF V=Vk(K3) GOTO 360
350     NEXT K3 : GOTO 370
360     Van=A(K3)+B(K3)*Q(V)+C(K3)*Q(V)^2
370     T=T+R(V)*Q(V)*ABS(Q(V))*SGN(Gozk(I,J))-Van
380   NEXT J
390   P(Ak(I))=-T
400 NEXT I
410 FOR I=1 TO Goz
420   FOR J=2 TO Goz(I)
430     V=ABS(Gozk(I,J))
440     P(V)=P(V)+P(Ak(I))*SGN(Gozk(I,J))
450   NEXT J
460 NEXT I
470 REM Optimal adim boyutunun hesabi
480 REM ******************************
490 A9=.0001 : H9=.1
500 X9(1)=A9 : X9=X9(1) : GOSUB 2000 : F9(1)=Z9
510 X9(2)=A9+H9 : X9=X9(2) : GOSUB 2000 : F9(2)=Z9
520 IF F9(2)(F9(1) GOTO 570
530 F9(3)=F9(2) : X9(3)=X9(2)
540 H9=H9/2 : X9(2)=A9+H9 : X9=X9(2) : GOSUB 2000 : F9(2)=Z9
550 IF F9(2))F9(1) GOTO 530
560 GOTO 600
```

EK-11 : (Devam ediyor)

```
570 H9=2*H9 : X9(3)=A9+H9 : X9=X9(3) : GOSUB 2000 : F9(3)=Z9
580 IF F9(3))F9(2) GOTO 600
590 X9(2)=X9(3) : F9(2)=F9(3) : GOTO 570
600 X=4*F9(2)-3*F9(1)-F9(3)
610 X=X*H9/(4*F9(2)-2*F9(3)-2*F9(1))
620 ; BLU 'X=';X
630 FOR I=1 TO Goz
640   FOR J=1 TO Goz(I)
650     V=ABS(Gozk(I,J))
660     Q(V)=Q(V)+X*P(Ak(I))*SGN(Gozk(I,J))
670   NEXT J
680 NEXT I
690 It=It+1 : IF It(2 GOTO 280
700 REM HARDY-CROSS ITERASYON TEKNIGI
710 REM ****************************
720 Dmax=0
730 FOR K2=1 TO Goz
740   Pay=0 : Pyd=0 : K1=Goz(K2)
750   FOR J=1 TO K1
760     Ka=Gozk(K2,J)
770     IF Ka(0 GOTO 790
780     C8=1 : GOTO 800
790     C8=-1 : Ka=-Ka
800     Q1=ABS(Q(Ka))
810     FOR L1=1 TO Vs
820       IF Ka=Vk(L1) GOTO 850
830     NEXT L1
840     Van=0 : Van1=0 : GOTO 870
850     Van=A(L1)+B(L1)*Q1+C(L1)*Q1^2
860     Van1=B(L1)+2*C(L1)*Q1
870     Pay=Pay+C8*(R(Ka)*Q(Ka)*Q1-Van)
880     Pyd=Pyd+(2*R(Ka)*Q1-Van1)
890   NEXT J
900   Dq=-Pay/Pyd
910   Dm=ABS(Dq)
920   IF (Dm-Dmax)(=0 GOTO 940
930   Dmax=Dm
940   FOR J=1 TO K1
950     Ka=Gozk(K2,J)
960     IF Ka)=0 GOTO 980
970     C8=-1 : Ka=-Ka : GOTO 990
980     C8=1
990     Q(Ka)=Q(Ka)+C8*Dq
1000  NEXT J
1010 NEXT K2
1020 It=It+1
1030 IF (Dmax-.005))0 GOTO 720
1040 ; #1 TAB(2) "KOL" TAB(7) "DIRENC" TAB(18) "DEBI"
1050 ; #1 "----------------------"
1060 FOR I=1 TO Kol
1070   ; #1 TAB(2) I TAB(7) R(I)*1000 TAB(16) Q(I)
1080 NEXT I
1090 D$=MID$(TIME$,11,9) : ; #1
1100 ; #1 "ITERASYON SAYISI=";It
1110 ; #1 "HESAPLAMA SURESI=";D$
1120 END
```