FİZ433 FİZİKTE BİLGİSAYAR UYGULAMALARI

(DERS NOTLARI)

Hazırlayan:

Prof.Dr. Orhan ÇAKIR

Ankara Üniversitesi, Fen Fakültesi, Fizik Bölümü

Ankara, 2017

İÇİNDEKİLER

1. LİNEER OLMAYAN DENKLEMLERİN KÖKLERİNİN BULUNMASI I/II

2. LİNEER DENKLEM SİSTEMLERİNİN ÇÖZÜLMESİ I/II

3. UYGUN EĞRİNİN BULUNMASI VE INTERPOLASYON I/II

4. SAYISAL İNTEGRAL HESAPLARI I/II

5. DİFERENSİYEL DENKLEMLERİN SAYISAL ÇÖZÜMLERİ I/II

6. BENZETİM I/II

7. FİZİKTE SEMBOLIK HESAPLAMA I/II

EKLER

KAYNAKLAR

KONU 2

LİNEER OLMAYAN DENKLEMLERİN KÖKLERİNİN BULUNMASI II

İkili Arama Yöntemi

Bu yöntem aralık yarılama veya ikili arama yöntemi olarak da bilinir. Yöntem x'in iki değeri $x=x_a$ ve $x=x_b$ ile başlar. Burada $f(x)$ fonksiyonunun bu aralıkta sürekli olduğu düşünülür. Eğer fonksiyonun bu noktalarda aldığı değerler arasında işaret farkı varsa bu durumda x_a ile x_b arasında bir yerde en az bir kök vardır denir. Bu yöntemde denklemin köklerini daha kolay bulabilmek için x_a ile x_b arası ikiye bölünür, Şekil 1.6. Bu iki aralıkta fonksiyonun işaret değiştirip değiştirmediği araştırılır.

Şekil 1.6 Kök bulmada aralık yarılama yöntemi

Burada orta nokta $x_m=(x_a+x_b)/2$ olarak tanımlanır, ve $f(x_m)$ de hesaplanır. Eğer $f(x_a)$ ve $f(x_m)$ zıt işaretli ise kök x_a ile x_m arasında olacaktır, diğer türlü x_m ile x_b arasında olacaktır. Böylece kökün bulunduğu bölge (x_b-x_a)'dan $(x_b-x_a)/2$'ye indirilmiş olur. Eğer kök sol yarı-aralıkta ise x_b'nin yeni değeri x_m olarak alınır. Aynı işlem n kez tekrarlandığında kökü içeren $(x_b-x_a)/2^n$ uzunluklu bir aralık elde edilir. Bu arama işlemi $|f(x_m)|<\varepsilon$ koşulu sağlanıncaya kadar devam eder, burada tolerans ε sıfıra yakın bir değerde alınır. Yakınsaklık testi için başka bir koşul da $|x_c-x_p|<|x_c\cdot\varepsilon_1|$

olarak tanımlanır. Burada x_c iterasyonda o andaki kök değeri, x_p ise bir önceki kök değeridir, ε_1 ise istenen tolerans değeridir.

- **Yöntemin Algoritması:**

1- program başlar

2- x_a ve x_b başlangıç değerleri alınır

3- x_m hesaplanır

4- $f(x_a)*f(x_m)>0$ ise $x_a=x_m$

5- $f(x_a)*f(x_m)<0$ ise $x_b=x_m$

6- $f(x_a)*f(x_b)$=tolerans ise kök x_m dir.

7- program sonlanır

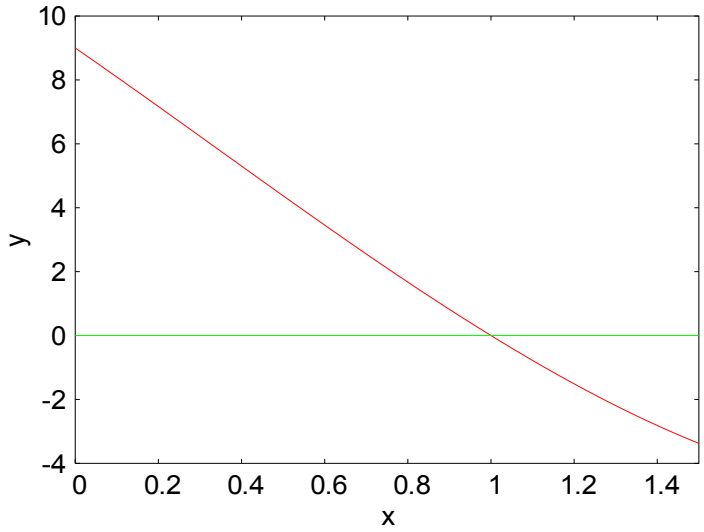
Örnek Problem: $f(x)=x^3-x^2-9x+9=0$ denkleminin köklerini ikili arama yöntemi ile bulunuz. Başlangıç değerlerini x_a=-2 ve x_b=1.5 alınız.

Çözüm:

x=-2 de $f(x)$=15.000

x=1.5 de $f(x)$=-3.375

Fonksiyonun bu hesaplanan değerleri arasında işaret farkı olduğundan denklemin en az bir kökü verilen x aralığı içerisindedir. Orta nokta x_m=(-2.0+1.5)/2=-0.25, ve bu noktada fonksiyon değeri $f(x)$=11.17185 olur. İncelendiğinde f(-0.25) ve f(1.5) zıt işaretlerde olduğu görülür o zaman kök bu x'lerin arasında olmalıdır. Bu durumda x_a=-0.25 ve x_b=1.5 alırız ve x_m=0.625 olur. Bunun gibi birkaç yinelemeden (iteration) sonra istenen tolerans değerine ulaşılmış olur ve köklerden birinin yaklaşık olarak x=0.999999 olduğunu buluruz, bu da kökün tam değeri x=1'e oldukça yakındır. Burada istenen kökün bölgesinin tahmini için grafiksel yöntem kullanılabilir. Kökün bulunduğu bölgede fonksiyonun düzgün davranışlı olması da yöntemin iyi sonuç vermesine neden olur. Fonksiyonun grafiği [0:1.5] aralığında çizildiğinde Şekil 1.7 elde edilir.



Şekil 1.7 $y=f(x)=x^3-x^2-9x+9$ fonksiyonunun 0:1.5 aralığında değişim grafiği

Aralık Yarılama Yöntemine Göre Kök Bulan FORTRAN Programı

Aşağıda verilen fortran programında girdi olarak *f(x)* fonksiyonu verilmelidir. Function *f(x)* bu girilecek fonksiyonu tanımlar, burada örnek olarak $f(x)=x^3-x^2-9x+9$ fonksiyonu verilmiştir. $f(x)=0$ denkleminin köklerinden biri belirli bir aralıkta aranacağından bunun da ana programda xa ve xb olarak girilmesi gerekmektedir. Kökleri hangi duyarlılıkta bulacağı programa tolerans (tol) olarak verilebilir.

- **FORTRAN programı**

```
program Arama2
xa=0.
xb=1.5
tol=1.E-06
dx=xb-xa
do while(abs(dx).gt.tol)
xm=(xa+xb)/2.
if(f(xa)*f(xm).lt.0.) then
xb=xm
dx=xb-xa
else
xa=xm
dx=xb-xa
endif
enddo
print*, " x= ",xm," dx=",dx
end

function f(x)
f=x*x*x-x*x-9.*x+9.
```

return
end

Programı derleyip çalıştırdıktan sonra elde edilecek sonuç x=0.999999 dir. Bu da x=1'de bir kökün olduğunu göstermektedir.

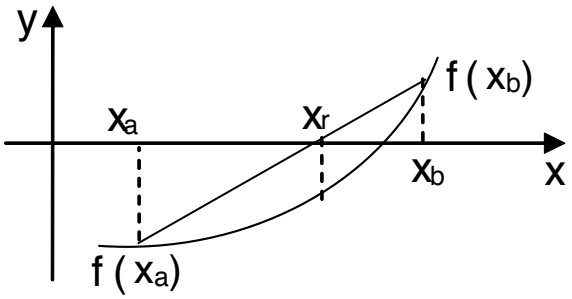
Bu yöntemde de başta büyük aralıklar seçmek birbirine yakın köklerin kaybedilmesine neden olabilir. Grafiksel olarak kökün bulunduğu bölge tahmin edildikten sonra bu yöntemin uygulanması daha uygun olacaktır.

Kiriş Yöntemi

Bu yöntem ikili arama yönteminin biraz değiştirilmiş şeklidir. Burada (x_a, x_b) aralığını ikiye bölüp orta nokta kullanmak yerine, [x_a,$f(x_a)$] noktasından [x_b,$f(x_b)$] noktasına bir doğru çizilir, bu doğrunun x-eksenini kestiği yer kökün yeni tahmini olur. Bu tahmin x_r ile gösterilirse

$$x_r = x_a + (x_b - x_a)\frac{f(x_a)}{f(x_a) - f(x_b)}$$

elde edilir. Bundan sonrası ikili arama yönteminin aynısıdır. Bu yöntem öncekinden daha hızlı yakınsamaktadır. Yakınsama testi olarak $|f(x_r)|<\varepsilon$ veya $|x_c-x_p|<|x_r\cdot\varepsilon_1|$ bağıntısı kullanılabilir. Burada x_c ve x_p sırasıyla, kökün o an hesaplanan değeri ve önceki adımda hesaplanan değerleridir. Burada tolerans ε_1 olarak alınmaktadır.



Şekil 1.8 Kiriş yönteminde kök bulma

- **FORTRAN programı**

program Kiris_Yontemi

```
a=-4.
b=0.
tol=1.E-06
call kiris(a,b,x,tol)
print*, "Kok= ",x
end

subroutine kiris(xa,xb,xr,tol)
xe=xa
iter=0
itermax=50
do while(iter.lt.itermax)
xy=xa+(xb-xa)*f(xa)/(f(xa)-f(xb))
iter=iter+1
xr=xy
if(abs(xy-xe).lt.abs(xy*tol)) return
if(f(xa)*f(xy).lt.0.) then
xb=xy
else
xa=xy
xe=xy
endif
enddo
return
end

function f(x)
```

```
f=x*x*x-x*x-9.*x+9.

return

end
```

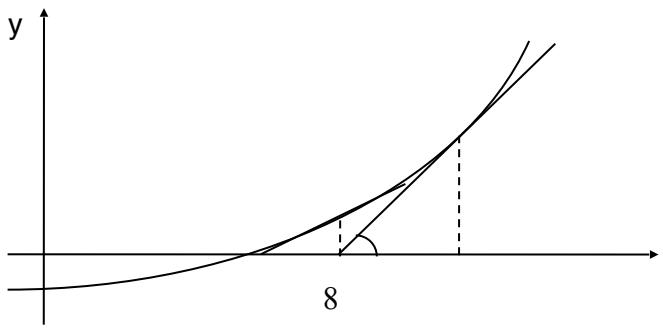
Ana program ve iki alt programdan (subroutine ve function) oluşan bu program çalıştırıldığında x=-4 ile x=0 arasında $f(x)$=0 denkleminin çözümünü bulmaktadır. Sonuç tol ile verilen duyarlılıkta x=-3 olarak bulunur.

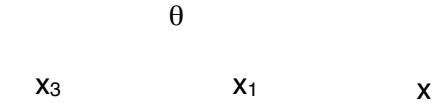
Newton-Raphson Yöntemi

Newton-Raphson yöntemi, en çok kullanılan kök bulma yöntemlerden biridir. Burada başta x_1 tahmini yapılır daha sonra bu noktada eğriye teğet çizilir bu teğetin x-eksenini kesme noktası belirlenir. Bu nokta ise ikinci tahmindir. Birinci tahmin bilindiğinde, ikinci tahmin

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

ile bulunur. Bu bağıntı tekrarlanma bağıntısı şeklinde kullanılarak gerçek kök hesaplanmasında iyi bir yaklaşıklık elde edilebilir.





Şekil 1.9 Newton-Raphson yönteminde kök bulma

Newton-Raphson yönteminin çok kabul görmesinin nedeni kök bulmada çok hızlı yakınsamasıdır. Yakınsaklık testi

$|f(x_{n+1})| < \varepsilon$

veya

$|x_{n+1}-x_n| < |x_{n+1} \cdot \varepsilon_1|$

koşullarına dayanmaktadır. Burada ε_1=1%-5% arasında olabilir. Önceki örnekteki denklem için bu yöntemde üçüncü tekrarlamadan sonra doğru sonuca ulaşılabilir.

Newton-Raphson Yöntemine Göre Kök Bulan FORTRAN Programı

Burada çözümü bulunacak denklemi $f(x)=e^x \ln x - x^2=0$ olarak alalım. Bu denklemin x=1 civarındaki kökünü bulalım. Kökler 10^{-6} mertebesindeki bir duyarlılığa kadar doğru bir şekilde bulunabilir. Ana program ve alt programlar ayrı ayrı derlenip sonra bir tek çalıştırılabilir dosya olacak şekilde bağlanabilir. Bunun için Linux işletim sisteminde makefile, Windows işletim sisteminde de .bat uzantılı toplu iş dosyaları yazılabilir, EK-1.2.

- **FORTRAN programı**

```
Program Newton_Raphson
a=1.
tol=1.e-06
```

```
      call newton(a,x,tol)

      write(*,20)x

20.   format(F10.6)
      end


      subroutine newton(a,x,tol)

      x0=a

      x1=x0

      do 10 while (abs(f(x1)).gt.tol)

      x1=x0-f(x0)/df(x0)

      x0=x1

      x=x0

10.   enddo
      return

      end


      function f(x)

      f=exp(x)*alog(x)-x*x

      return

      end


      function df(x)

      df=exp(x)*(alog(x)+1./x)-2.*x

      return

      end
```

Program çalıştırıldıktan sonra ekranda x=1.694601 kök değeri elde edilir.

Newton-Raphson yöntemi çoğu durumda verimli bir şekilde kullanılabilir. Ancak bazı durumlarda denklemin çözümünü bulmak zorlaşır. Böyle durumlara örnekler: çok kök

olması durumu, fonksiyonun şekli gereği bir yansıma noktasının ($f''(x)=0$) bulunması, bir maksimum veya minimum etrafında salınım, sıfır eğimin ($f'(x)=0$) bulunduğu bölge durumu. Bu durumlardan kurtulmanın yolu, genellikle kökün bulunduğu bölgenin grafik yöntemi ile belirlenmesi ve başlangıç tahmin değerinin değiştirilmesidir.

Sekant Yöntemi

Bu yöntem Newton-Raphson yönteminde görülen bir probleme yaklaşıklıkla çözüm getirmektedir. $f(x)$ fonksiyonunun türevini hesaplamada iki ardışık işlevsel yaklaşıklık kullanılmıştır. Doğrunun x_n noktasında eğimi

$$\text{Egim} = f'(x_n) = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}$$

ile verilir. Sonraki noktanın yeri

$$x_{n+1} = \frac{x_{n-1} f(x_n) - x_n f(x_{n-1})}{f(x_n) - f(x_{n-1})}$$

ile bulunabilir. Sekant yönteminde başta iki kök tahmini ile başlanır (x_0 ve x_1), x_2 ise interpolasyon ile bulunur. Örnek fonksiyon $f(x)=e^x \ln x - x^2$ alalım ve $x=1$ ile $x=2$ arasındaki kökünü bulalım. Bunun için yazılabilecek bir FORTRAN programı aşağıda verilmiştir.

- **FORTRAN programı**

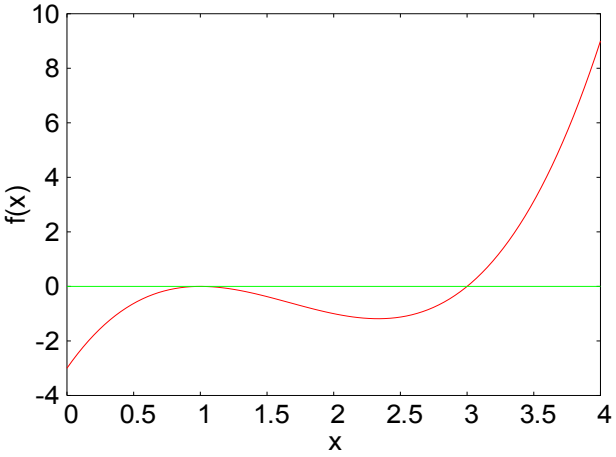
```
program Sekant_Yontemi
a=1.
b=2.0
dx=(b-a)/10.
x0=(a+b)/2.
tol=1.e-06
call sekant(tol,x0,dx,istep)
write(*,9) istep,x0,dx
```

```
 9    format(I4,2F10.6)
      end

      subroutine sekant(tol,x0,dx,istep)
      istep=0
      x1=x0+dx
      do 10 while(abs(dx).gt.tol)
      d=f(x1)-f(x0)
      x2=x1-(x1-x0)*f(x1)/d
      x0=x1
      x1=x2
      dx=x1-x0
      istep=istep+1
10    enddo
      return
      end

      function f(x)
      f=exp(x)*alog(x)-x*x
      return
      end
```

Program çalıştırıldığında 5 iterasyondan sonra ekranda kök değeri x=1.694601 elde edilir. Sekant yöntemi aralık yarılama yönteminden daha verimli, fakat Newton-Raphson yönteminden daha az verimlidir. Bunun nedeni birinci mertebe türev için iki noktalı formulü kullanmasıdır. Bazı durumlarda fonksiyonun türevinin analitik hesaplanması zor olabilir bu durumda sekant yöntemi çok kullanışlı olmaktadır.

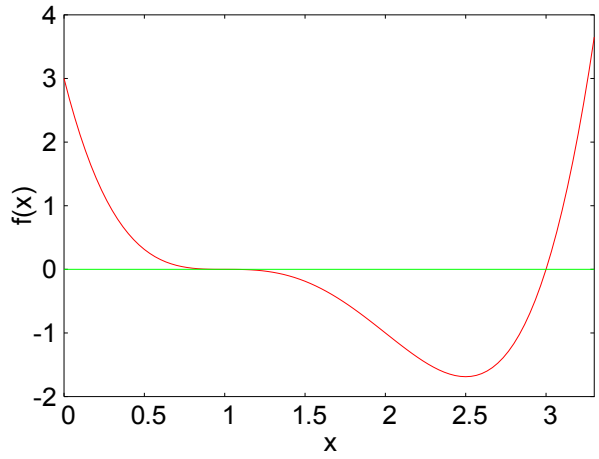
Katlı Köklerin Bulunması

Bir katlı kök, fonksiyonun x-eksenine teğet olduğu noktaya karşı gelir. Örneğin, bir çift kök $f(x)=(x-3)(x-1)(x-1)=x^3-5x^2+7x-3=0$ denkleminden elde edilebilir, Şekil 1.10.



Şekil 1.10 İki eşit kök olması durumu

Grafikten görüldüğü gibi, eğri x=1 noktasında x-eksenine teğet hale gelmekte ve kökün bulunduğu yerde x-eksenini kesmemektedir. Bir başka örnek de, üç katlı kök için $f(x)=x^4-6x^3+12x^2-10x+3$ fonksiyonun grafiğinden görülebilir, Şekil 1.11.



Şekil 1.11 Üç eşit kök içeren bir fonksiyonun grafiği

Şekil 1.11 de gösterilen fonksiyonun kök değerinde x-eksenine teğet olduğu, fakat bu durumda ekseni kestiği görülmektedir. Genellikle fonksiyonun, tek sayıda eşit kökü olması durumunda x-eksenini keser, çift sayıda eşit kökü olması durumunda ise x-eksenini kesmez.

Çok katlı kökler, sayısal yöntemler için bazı zorluklar çıkarmaktadır. Karşılaşılan problemler ve çözüm yolları aşağıda verilmiştir:

- Fonksiyon çoklu köklerin bulunduğu yerde işaret değiştirmeyebilir. Bu durumda daha güvenilir yöntemler uygulanmalıdır.
- Diğer bir problem de hem fonksiyonun hem de fonksiyonun türevinin kök değerinde sıfıra gitmesidir. Bu problem Newton-Raphson ve Sekant yöntemlerini etkilemektedir. Bu problemden kurtulmak için $f(x)$ fonksiyonunun daima $f'(x)$ fonksiyonundan daha önce sıfıra gittiği gerçeğinden faydalanılır, programda fonksiyonun sıfıra gittiği kontrol edilerek, hesaplama $f'(x)$ sıfıra gitmeden sonlandırılır.
- Katlı kökler için, Newton-Raphson ve Sekant yöntemleri lineer yakınsaktır. Bunu karesel yakınsaklığa çevirmek için formulasyonda bir değişiklik yapılır. Yeni bir fonksiyon $u(x) = f(x)/f'(x)$ tanımlanır, bu fonksiyon da orijinal fonksiyon gibi aynı yerde köklere sahiptir. Newton-Raphson yönteminin başka bir formu elde edilir:

  $$x_{i+1} = x_i - \frac{u(x_i)}{u'(x_i)} = x_i - \frac{f(x_i)f'(x_i)}{\left[f'(x_i)\right]^2 - f(x_i)f''(x_i)}$$

  burada ikinci eşitliğin sağ tarafı elde edilirken $u'(x)$ türevi yerine konulmuştur.

Lineer Olmayan Denklem Sistemleri

Buraya kadar bir tek denklemin köklerinin bulunması ile ilgilenmiştik. Bu alt bölümde ise eşzamanlı lineer olmayan denklem sisteminin köklerinin bulunması ile ilgileneceğiz. Bu denklem sistemi aşağıdaki gibi yazılabilir:

$$\begin{aligned} f_1(x_1, x_n, \cdots, x_n) &= 0 \\ f_2(x_1, x_n, \cdots, x_n) &= 0 \\ \vdots \qquad\qquad & \vdots \\ f_n(x_1, x_n, \cdots, x_n) &= 0 \end{aligned}$$

Bu denklem sisteminin çözümü, denklemleri sağlayan x değerlerinin bir kümesinden oluşur. Örnek olarak iki denklemden oluşan bir lineer olmayan denklem sistemi düşünelim:

$$u(x,y) = x^2 + xy - 15$$
$$v(x,y) = y + 3xy^2 - 38$$

Burada çözüm, $u(x,y)$ ve $v(x,y)$ fonksiyonlarını sıfır yapan x ve y değerleridir. Bu denklem sisteminin sayısal çözülmesi için en çok kullanılan iki yöntemden bahsedebiliriz. Bunlar, sabit-nokta iterasyonu ve Newton-Raphson yöntemidir.

- ## Sabit-nokta iterasyonunun kullanılması:

$x^2 + xy - 15 = 0$ ve $y + 3xy^2 - 38 = 0$ denklem sisteminin başlangıç tahminlerini x=1 ve y=1 alarak köklerini bulalım. Öncelikle denklemleri $x = \sqrt{15 - xy}$ ve $y = \sqrt{(38 - y)/3x}$ olarak yeniden yazalım. Burada eşitliklerin sağ taraflarında x ve y için verilen ilk değerleri kullanarak, eşitliklerin sol taraflarındaki yeni x ve y değerlerini bulalım. Her iterasyonda yeni değerler bulmak için bir öncekileri kullanırsak bir kaç iterasyonda kökler için doğru çözüme ulaşmış oluruz. Bu işlemler serisinin FORTRAN programı aşağıda verilmiştir.

- **FORTRAN programı**

```
Program Sabit_Nokta_Iterasyon
x=1.
y=1.
iter=0
do i=1,10
x=sqrt(15.-x*y)
y=sqrt((38.-y)/(3.*x))
iter=iter+1
write(*,*)"Iter.=",iter,"  Kokler= ",x,y
enddo
end
```

Program çalıştırıldığında 8 iterasyondan sonra gerçek değerlere ulaşılmaktadır, ve denklem sisteminin kökleri x=3 ve y=2 olarak bulunmaktadır. Bu yöntemin yakınsaklığı

$$\left|\frac{\partial u}{\partial x}\right| + \left|\frac{\partial v}{\partial x}\right| < 1 \quad \text{ve} \quad \left|\frac{\partial u}{\partial y}\right| + \left|\frac{\partial v}{\partial y}\right| < 1$$

koşulları ile sağlanır. Bu koşullar lineer olmayan denklem sistemlerinin çözümünde çok sınırlı bir kullanıma sahiptir. Ancak lineer denklem sistemlerinin çözümünde kullanılması çok faydalı olabilir.

İterasyon	x	y
1	3.7416575	1.81554997
2	2.8647573	2.0519011
3	3.02023196	1.99185252
4	2.99735618	2.00110817
5	3.00032759	1.99986005
6	2.9999609	2.00001693
7	3.00000453	1.99999797
8	2.99999952	2.00000024
9	3.	2.
10	3.	2.

- **Newton-Raphson yönteminin kullanılması:**

Bulunmak istenen kökün, verilen başlangıç tahminine (x_i) uygun olarak bulunan x_{i+1} noktası, eğimin x-eksenini kestiği yerdeki noktadır. Burada tek denklemli Newton-Raphson yöntemini genişleterek, çok değişkenli Taylor serisi açılımından da faydalanarak lineer olmayan denklem sistemlerinin çözümlerini bulabiliriz. İki değişkenli lineer olmayan denklem sitemi için birinci-mertebe Taylor serisi

$$u_{i+1} = u_i + (x_{i+1} - x_i)\frac{\partial u_i}{\partial x} + (y_{i+1} - y_i)\frac{\partial u_i}{\partial y}$$

$$v_{i+1} = v_i + (x_{i+1} - x_i)\frac{\partial v_i}{\partial x} + (y_{i+1} - y_i)\frac{\partial v_i}{\partial y}$$

olarak yazılabilir. Burada x ve y kökleri, u_{i+1} ve v_{i+1} 'in sıfır olduğu değerlerdir. Bunun için denklemler yeniden düzenlenerek x_{i+1} ve y_{i+1} için çözülürse

$$x_{i+1} = x_i - \frac{u_i \dfrac{\partial v_i}{\partial y} - v_i \dfrac{\partial u_i}{\partial y}}{\dfrac{\partial u_i}{\partial x}\dfrac{\partial v_i}{\partial y} - \dfrac{\partial u_i}{\partial y}\dfrac{\partial v_i}{\partial x}}$$

$$y_{i+1} = y_i - \frac{v_i \dfrac{\partial u_i}{\partial x} - u_i \dfrac{\partial v_i}{\partial x}}{\dfrac{\partial u_i}{\partial x}\dfrac{\partial v_i}{\partial y} - \dfrac{\partial u_i}{\partial y}\dfrac{\partial v_i}{\partial x}}$$

elde edilir. Bu denklemler Newton-Raphson yönteminin iki denklem biçimidir. Bu denklemlerin paydası sistemin Jacobian determinantıdır. Bu yöntemle denklem sisteminin kökleri yine iterasyon tekniği kullanılarak bulunabilir.

Örnek denklem sistemi $u(x,y) = x^2 + xy - 15$ ve $v(x,y) = y + 3xy^2 - 38$ için başlangıç tahminlerini x=1 ve y=1 alarak köklerini bulalım. Öncelikle fonksiyonların türevlerini hesaplayalım:

$$\frac{\partial u}{\partial x} = 2x + y = 3\,; \quad \frac{\partial u}{\partial y} = x = 1\,; \quad \frac{\partial v}{\partial x} = 3y^2 = 3\,; \quad \frac{\partial v}{\partial y} = 1 + 6xy = 7$$

Jacobian determinantının değerini 18 olarak hesaplarız. Buradan fonksiyonların başlangıç değerleri $u(1,2)$=-13 ve $v(1,2)$=-34 dir. Bu hesaplanan değerler Newton-Raphson yönteminin iki-denklem biçiminde yerine konulursa x≈4.2 ve y≈4.5 elde edilir. Bu sonuçlar gerçek değerlerden (x,y)=(3,2) biraz uzaktır. İkinci iterasyonda x≈2.9 ve y≈3.2 elde edilir. Bu hesaplama istenen bir duyarlılık elde edilinceye kadar tekrarlanabilir. Bu problemin çözümü ile ilgili FORTRAN programı aşağıda verilmiştir.

- **FORTRAN programı**

```
Program N_R_1
x=2.
y=3.
iter=0
open(1,file="N_R1.txt")
do i=1,10
call fonk(x,y,u,v,dudx,dudy,dvdx,dvdy)
xjd=dudx*dvdy-dudy*dvdx
x=x-(u*dvdy-v*dudy)/xjd
y=y-(v*dudx-u*dvdx)/xjd
iter=iter+1
write(1,*)"Iter.=",iter,"  Kokler= ",x,y
write(*,*)"Iter.=",iter,"  Kokler= ",x,y
enddo
end

subroutine fonk(x,y,u,v,dudx,dudy,dvdx,dvdy)
u=x**2+x*y-15.
v=y+3.*x*y**2-38.
dudx=2.*x+y
dudy=x
dvdx=3.*y**2
dvdy=1.+6.*x*y
return
```

```
end
```

Program çalıştırıldığında verilen denklem sisteminin köklerini aşağıdaki gibi iterasyona bağlı olarak buluruz.

İterasyon	X	y
1	4.16666651	4.5
2	2.93557215	3.22317481
3	2.90921521	2.25364256
4	2.99519849	2.00823355
5	3.00000405	1.99999118
6	3.	2.
7	3.	2.

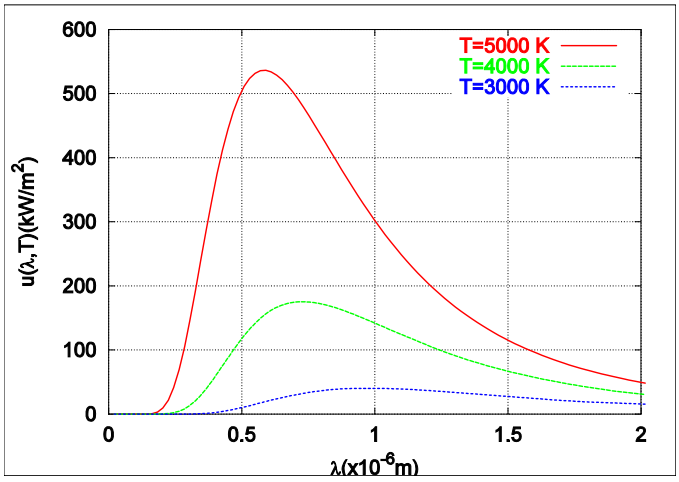
Burada elde edilen sonuçlara göre, verilen iki-denklem sisteminde Newton-Raphson yöntemi, sabit-nokta iterasyonuna göre daha hızlı yakınsamaktadır. Başlangıç değerleri de fiziksel sistemin özelliklerine göre tahmin edilebilir.

İki-denklemli sistemin çözüm yöntemleri n-denklemli sisteme genişletilebilir. Bu durumda eşzamanlı denklem sistemi çözümü, matris cebiri içeren yöntemlerle çok daha verimli olarak yapılabilir.

Fizikte Uygulamalar

Örnek 1: Siyah cisim, üzerine düşen her dalga boyundaki ışımayı soğuran bir cisim olarak tanımlanabilir. Bu olayı anlamak için içi boş bir kürenin iç yüzeyi siyahla kaplı olduğunu ve kürenin üzerinde de küçük bir delik olduğunu düşünelim. Bu delikten içeriye giren ışınımın bir daha dışarı çıkamadığını kabul ediyoruz. Siyah yüzey tarafından soğrulan ışınım cisim ile etkileşip sonunda bir ısısal denge kurulduğunda, siyah cismin yaptığı ışımanın spektrumu deneysel olarak ölçülebilir. Bu ışımanın birim yüzeye düşen ışıma gücü $dP=p(\lambda,T)d\lambda$ ile verilir. Burada $p(\lambda,T)$, dalgaboyu λ ile $\lambda+d\lambda$ arasında olan ışıma gücü yoğunluğudur. Deneysel olarak gözlenen bu yoğunluk eğrisi Şekil 1.12 de gösterilmiştir. Siyah cisim ışımasının gözlenen bu spektrumu klasik fizik yasaları ile tam olarak açıklanamamıştır. 1900 yılında Max Planck, ışımanın kuantum yapısını (atomların ışıma yoluyla enerji alış verişleri sürekli değil, kesikli

spektrumlar yoluyla gerçekleşir) öngören bir varsayımla, bu eğriyi tam olarak açıkladı. Bu formulasyon, kısa dalgaboyu bölgesinde geçerli olan Wien formulu (1893) ile uzun dalgaboyu bölgesinde geçerli olan Rayleigh-Jeans (1900) yasasını birleştirdi. Planck dağılımının bazı özellikleri şunlardır:



Şekil 1.12 Siyah cisim ışımasında ışıma gücünün dalgaboyuna göre değişimi

(i) toplam ışıma enerjisi a bir sabit olmak üzere aT^4 ile verilir, Stefan-Boltzman bağıntısı,
(ii) enerji yoğunluğunun maksimum olduğu bir dalgaboyu ile denge sıcaklığı arasında, b bir sabit olmak üzere, $\lambda_m T=b$ bağıntısı vardır, Wien kayma yasası.

Bu örnekte Wien kayma yasasını sayısal olarak sağlamak için Planck formülünü

$$p(\lambda) = \frac{8\pi hc}{\lambda^5}\frac{1}{e^{hc/\lambda kT}-1}$$

bir boyutsuz değişken cinsinden yazalım ve bu niceliğe göre türevini alıp sıfıra eşitleyelim. Boyutsuz değişkeni $x=hc/\lambda kT$ alırsak, $p(x)$ ve türevi

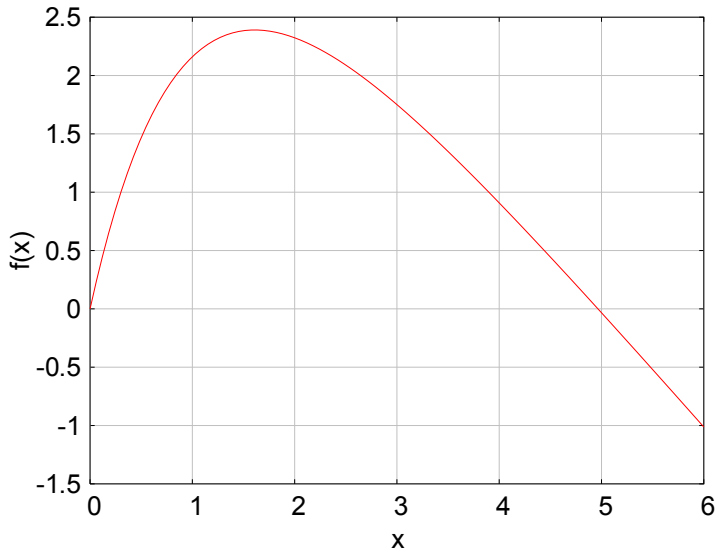
$$p(x) = A\frac{x^5}{e^x - 1}$$

$$\frac{dp(x)}{dx} = A\frac{5x^4(e^x - 1) - x^5 e^x}{(e^x - 1)^2} = 0$$

$$\Rightarrow (5 - x) - 5e^{-x} = 0$$

şeklinde yazılabilir. Bu denklemin analitik çözümü olmadığından, sayısal yöntemlerden sekant yöntemini uygulayan bir program yazınız, denklemin kökünü bulunuz ve Wien kayma yasasını sayısal olarak elde ediniz.

Çözüm 1: Verilen denklemin çözümünden x_m ve daha sonra λ_m bulunabilir. Burada $f(x)=(5-x)-5e^{-x}$ fonksiyonu incelendiğinde ilk terimin x>3 için negatif olduğunu üstel terimin de daima 3'den küçük olduğunu görürüz, Şekil 1.13. Burada x=0 değeri denklemi sağlar fakat fiziksel değildir. Buna göre [1:6] aralığında kök aranabilir. Probleme sayısal yöntemlerden sekant yöntemini uygulayalım, bu durumda ilgili FORTRAN programı aşağıda verilmiştir.



Şekil 1.13 $f(x)=(5-x)-5e^{-x}$ fonksiyonunun grafiği

- FORTRAN programı

```
program S_Y_2

data h,c,bk/6.626E-34,2.998E+08,1.381E-23/
```

```
      a=1.
      b=6.0
      dx=(b-a)/10.
      x0=(a+b)/2.
      tol=1.e-06
      istep=0
      call sekant(tol,x0,dx,istep)
      sabit=h*c/(bk*x0)
      write(*,9)istep,x0,sabit
9     format(I4,2F10.6)
      end

      subroutine sekant(tol,x0,dx,istep)
      x1=x0+dx
      do 10 while(abs(dx).gt.tol)
      d=f(x1)-f(x0)
      x2=x1-(x1-x0)*f(x1)/d
      x0=x1
      x1=x2
      dx=x1-x0
      istep=istep+1
10    enddo
      return
      end

      real function f(x)
      f=5.-x-5.*exp(-x)
```

```
return

end
```

Programda Planck sabiti h=6.626×10^{-34} Js, ışık hızı c=2.998×10^{8} m/s ve Boltzmann sabiti k=1.381×10^{-23} J/K olarak girilmiştir, (Eidelman et al., 2004). Program çalıştırıldığında kök değeri x_m=4.965114 ve sabit değeri b=0.002897 mK bulunur. Bu sonuçlardan da Wien yasasını ($\lambda_m T=b=hc/kx_m$) sayısal olarak elde etmiş olduk. Buradan da sıcaklığı T=5000 K alırsak maksimum dalgaboyunu λ_m=0.5794 μm olarak buluruz. Görünür ışığın dalga boyu 0.4-0.8 μm arasındadır.

Örnek 2: Paraşüt problemi. Kütlesi 60 kg olan bir paraşütçü havada durgun halde iken balondan aşağı atlıyor. Paraşütü açmadan t=10 s sonra düştükten sonra hızı 40 m/s oluyor, paraşütçü için sürtünme katsayısını grafik yöntemi ve Newton-Raphson yöntemi ile hesaplayınız.

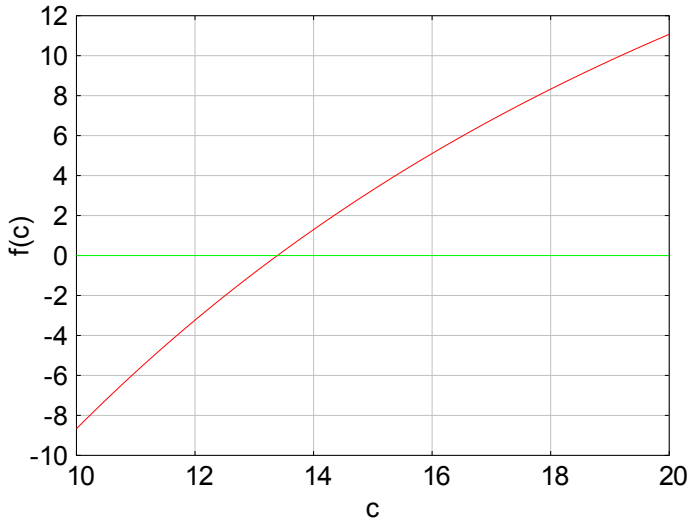
Çözüm 2: M kütleli bir paraşütçü düşmeye başladığında üzerine iki kuvvet etki eder. Aşağı doğru yer çekimi kuvveti (F_A=Mg) ve yukarı doğru hava sürtünme kuvveti (F_Y=-cv). Burada g≈10 m/s^2 yer çekim ivmesi, c sürtünme katsayısı ve v de paraşütçünün düşey hızıdır. Newton'un ikinci yasasına göre hareket denklemi

$$\frac{dv}{dt}=\frac{F_A+F_Y}{M}=g-\frac{c}{M}v$$

ile verilir. Paraşütçünün başlangıçta (t=0 anında) durgun (v=0) olduğunu düşünürsek bu denklemin analitik çözümü

$$v(t)=\frac{gM}{c}(1-e^{-ct/M})$$

olarak elde edilir. Burada ivme $a(t)=ge^{-ct/M}$ bulunur. Sayısal değerler yerine konursa f(c)=40-600(1-e$^{-c/6}$)/c=0 denklemine ulaşılır. Bu fonksiyonun grafiği çizdirildiğinde Şekil 1.14,



Şekil 1.14 $f(c)=40-600(1-e^{-c/6})/c$ fonksiyonunun grafiği

istenen kök değerinin 12 ile 14 arasında olduğu görülür. Bu denklem sayısal yöntemlerle de çözülebilir. Newton-Raphson yöntemini uygulayabilmek için bu fonksiyonun türevine de ihtiyaç duyulmaktadır, bunun için $f'(c)=600/c^2-(100/c)(6/c-1)e^{-c/6}$ olarak hesaplanır.

- **FORTRAN programı**

```
Program Parasut
a=1.
tol=1.e-06
call newton(a,c,tol)
write(*,*) " c= ",c
end

subroutine newton(a,x,tol)
x0=a
x1=x0
do 10 while (abs(f(x1)).gt.tol)
x1=x0-f(x0)/df(x0)
```

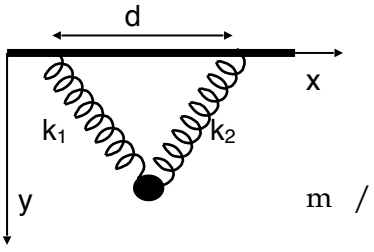
```
      x0=x1
      x=x0
10.   enddo
      return
      end

      function f(x)
      f=40.-600.*(1.-exp(-x/6))/x
      return
      end

      function df(x)
      df=600./x**2-100./x*(6./x-1.)*exp(-x/6.)
      return
      end
```

Program çalıştırıldıktan sonra kök değeri c=13.3896713 olarak bulunur.

Örnek 3: İki boyutlu kütle-yay sistemi Şekil 1.15 de gösterilmiştir. Potansiyel enerji

$$V(x,y)=\frac{1}{2}k_1\left(\sqrt{x^2+y^2}-l_1\right)^2+\frac{1}{2}k_2\left(\sqrt{(x-d)^2+y^2}-l_2\right)^2-mgy$$

ile verilir. Burada k_1 ve k_2 yay sabitleri; l_1 ve l_2 yayların serbest uzunlukları; m cismin kütlesi ve g çekim ivmesidir. Statik denge durumunda $\mathbf{F}=-\nabla V=0$ dır, bu durumda x ve y konumlarını bulmak için Newton-Raphson yöntemini kullanan bir program yazınız. Bilinenler $k_2=3k_1=30$ N/m ; $l_1=l_2=0.1$ m; $d=0.1$ m; $m=0.1$ kg; $g=9.81$ m / s^2.



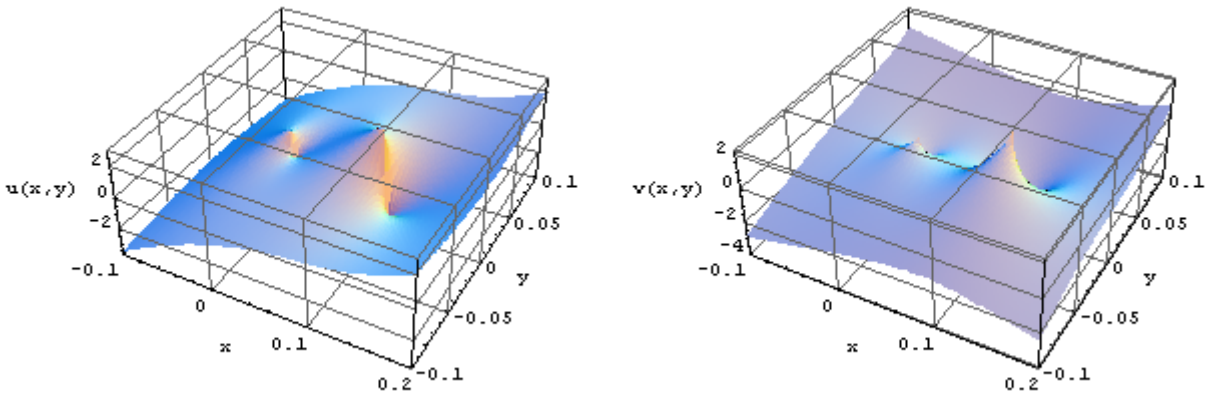
Şekil 1.15 Kütle-yaylar sistemi

Çözüm 3: Statik denge durumu için $\mathbf{F}=-\nabla V=0$ sağlanmalıdır. Problemde $k_2>k_1$ olduğundan Şekil 1.15'deki düzenlenimde fiziksel olarak x ve y değerlerinin pozitif olmasını bekleriz.

$$\frac{\partial V}{\partial x} = u = \frac{k_1\left(\sqrt{x^2+y^2}-l_1\right)x}{\sqrt{x^2+y^2}} + \frac{k_2\left(\sqrt{(x-d)^2+y^2}-l_2\right)(x-d)}{\sqrt{(x-d)^2+y^2}}$$

$$\frac{\partial V}{\partial y} = v = \frac{k_1\left(\sqrt{x^2+y^2}-l_1\right)y}{\sqrt{x^2+y^2}} + \frac{k_2\left(\sqrt{(x-d)^2+y^2}-l_2\right)y}{\sqrt{(x-d)^2+y^2}} - mg$$

Burada u ve v fonksiyonlarının sıfır değerlerini aldığı x ve y değerlerini bulmak istediğimizden bu fonksiyonların 3-boyutlu grafiklerini çizebiliriz, Şekil 1.16.



Şekil 1.16. $u(x,y)$ ve $v(x,y)$ fonksiyonlarının 3-boyutlu grafikleri

Bu grafiklerden görüldüğü gibi kök değerleri 0.1 civarında olmalıdır. Newton-Raphson yöntemini kullanmak için u ve v nin türevleri hesaplanırsa,

$$\frac{\partial u}{\partial x} = \frac{k_1\left((x^2+y^2)^{3/2} - l_1 y^2\right)}{(x^2+y^2)^{3/2}} + \frac{k_2\left(((x-d)^2+y^2)^{3/2} - l_2 y^2\right)}{((x-d)^2+y^2)^{3/2}}$$

$$\frac{\partial u}{\partial y} = \frac{y(-dk_2 l_2 + \frac{x(k_2 l_2 (x^2+y^2)^{3/2} + k_1 l_1 ((x-d)^2+y^2)^{3/2})}{(x^2+y^2)^{3/2}})}{((x-d)^2+y^2)^{3/2}}$$

$$\frac{\partial v}{\partial x} = \frac{y(-dk_2 l_2 + \frac{x(k_2 l_2 (x^2+y^2)^{3/2} + k_1 l_1 ((x-d)^2+y^2)^{3/2})}{(x^2+y^2)^{3/2}})}{((x-d)^2+y^2)^{3/2}}$$

$$\frac{\partial v}{\partial y} = \frac{k_1\left((x^2+y^2)^{3/2} - l_1 x^2\right)}{(x^2+y^2)^{3/2}} + \frac{k_2\left(((x-d)^2+y^2)^{3/2} - l_2 d^2 + 2dl_2 x - l_2 x^2\right)}{((x-d)^2+y^2)^{3/2}}$$

elde edilir. Bu ifadeler yöntemin formullerinde yerine yazılırsa x ve y çözümleri elde edilir. Bu problem için FORTRAN alt programı aşağıda verilmiştir. Burada gerekli ana program önceki bölümde verilmiştir.

- **FORTRAN alt programı**

```
subroutine fonk(x,y,u,v,dudx,dudy,dvdx,dvdy)
xm=0.1
xg=9.81
xd=0.1
xl1=0.1
xl2=0.1
xk1=10.
xk2=30.
xy=x**2+y**2
xdy=(x-xd)**2+y**2
u=(xk1*(sqrt(xy)-xl1)*x/sqrt(xy)
. +xk2*(sqrt(xdy)-xl2)*(x-xd)/sqrt(xdy))
v=(xk1*(sqrt(xy)-xl1)*y/sqrt(xy)
```

```
. +xk2*(sqrt(xdy)-xl2)*y/sqrt(xdy)-xm*xg)
 dudx=xk1*(xy*sqrt(xy)-xl1*y**2)/(xy*sqrt(xy))+
.   xk2*(xdy*sqrt(xdy)-xl2*y**2)/(xdy*sqrt(xdy))
 dudy=y*(-xd*xk2*xl2+(x*(xk2*xl2*xy*sqrt(xy))+
.   xk1*xl1*xdy*sqrt(xdy))/(xy*sqrt(xy)))
.   /(xdy*sqrt(xdy))
 dvdx=dudy
 dvdy= xk1*(xy*sqrt(xy)-xl1*x**2)/(xy*sqrt(xy))+
. xk2*(xdy*sqrt(xdy)-xl2*xd**2
. +2.*xd*xl2*x-xl2*x**2)/(xdy*sqrt(xdy))
 return
 end
```

Problemin salınımlı doğası gereği, program çalıştırıldıktan sonra ancak 60 iterasyonda x= 0.0676503 m ve y=0.118597 m sabit değerleri elde edilmektedir. Bu sonuçlar başlangıç değerleri x=y=0.1 m ile elde edilmiştir. Başka değerler ile başlarsak daha farklı iterasyon değerlerinde yine bu sonuçları elde edebiliriz

ÖZET

Lineer olmayan bir denklemin kökleri: f(x)=0 denklemi çözülerek x değerleri bulunur. Bu uygulamanın grafiksel anlatımı aşağıdaki şekilde verilmiştir.