Bölüm 1

Sayısal Analiz Süreci

Bu bölümde matematiksel analiz yöntemleri olarak bilinen *analitik*, *sayısal*, *kalitatif* ve *sembolik analiz* yöntemlerini kısaca tanıtarak, *sayısal analiz* yönteminin bütün aşamalarını üç temel düzey örnek üzerinde inceliyoruz.

1.1 Matematiksel analiz yöntemleri

Çözümü matematiksel yöntemlerin kullanılmasını gerektiren bir problemin, ilgili matematiksel yöntemler yardımıyla incelenmesi veya diğer bir deyimle analiz edilmesi *matematiksel analiz* olarak adlandırılmaktadır. Herhangi bir matematiksel problem *analitik, sayısal, kalitatif* ve *sembolik analiz* adı verilen analiz türlerinden birisi veya birden fazlası yardımıyla analiz edilebilir.

Analitik analiz, verilen problemi

- teorik olarak geliştirilmiş matematiksel formülasyon veya argümanlar yardımıyla ve
- kağıt-kalem gibi temel araçları kullanarak irdeleme sürecidir.

Analitik analiz yöntemi, verilen problem için akla gelen yöntemdir, ancak her problem için ve özellikle son yüzyılda uygulamalı matematik kapsamında incelenmesi gereken problemlerin çözümü veya analizi için yeterli değildir.

Konuya ışık tutması açısından aşağıda sunulan ve bir kısmı ortaöğretim matematik ders müfredatlarında ve diğer bir kısmı ise elemanter lisans derslerinde incelenen temel konulardan bir kaçına göz atalım:

1. $a, b, c \in R$ ve $a \neq 0$ olmak üzere $ax^2 + bx + c = 0$ denkleminin köklerini belirlenmesi,

2. $(x_0, y_0), (x_1, y_1)$ noktalarından geçen doğru denkleminin belirlenmesi

3. $a_{ij} \in R, b_i \in R$ olmak üzere

$$\begin{aligned} a_{11}x + a_{12}y &= b_1 \\ a_{21}x + a_{22}y &= b_2 \end{aligned}$$

   denklem sisteminin çözümü,

4. $\int_0^1 \sin(x)dx$ integralinin hesaplanması ve

5. $y' = t - y, t \in (a, b), y(a) = y_0$ başlangıç değer probleminin çözümünün belirlenmesi problemini gözönüne alalım.

Birinci problem üzerinde uzun yıllar öncesinden beri çalışmış bir problemdir.

$$3x^2 + 5x - 1 = 0$$

gibi bazı özel denklem türlerinin çözümü için geometrik yaklaşımlar 800 lü yılların başında Harezmi[1] tarafından geliştirilmiştir. Katsayıları verilen en genel ikinci dereceden polinomun denklemin çözümünü veren

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

formülü(analitik yöntemi) mevcuttur ve bu formül kullanılarak problemin kağıt-kalem çözümü(kağıt üzerinde ve kalem yardımıyla) elde edilebilir. O halde bu problemin analitik yöntemle analizi mümkündür. Ancak şimdi de

$$ax^3 + bx^2 + cx + d = 0$$

veya

$$ax^4 + bx^3 + cx^2 + dx + e = 0$$

[1] 780(Horasan)-850(Bağdat), Cebir ismi yazmış olduğu ve "El Cebri" sözcüğünü içeren kitabından türetilmiştir[8]

denklemlerinin köklerini bulmaya çalışalım. Bu durumda 3. veya 4. dereceden polinomların köklerini bulmamız gerekmektedir. Söz konusu denklem köklerinin belirlenmesi için de Hayyam [2] ve/veya Cardano[3]ya atfedilen karmaşık formüller mevcuttur, ve bu formüller kağıt üzerinde ve kalem yardımıyla ilgili köklerin belirlenmesi için genelde kullanılamazlar. Bununla beraber derecesi beşe eşit veya beşten daha büyük olan en genel bir polinomun köklerinin belirlenmesi için hiç bir analitik çözüm yönteminin verilemeyeceğini ise Abel[4] ve Galois[5] isimli matematikçilerin çalışmalarından biliyoruz.

Öte yandan *transandant* fonksiyon olarak bilinen üstel, logaritmik, trigonometrik fonksiyonlar içeren denklemlerin çözümü için herhangi bir genel formül söz konusu değildir. Örneğin

$$e^x - (x+4) = 0$$

denklemini sağlayan ve birisi negatif($x \doteq -3.9813$) diğeri ise pozitif($x \doteq 1.7490$) olan x değerlerini belirlemek için bir formül mevcut değildir. En genel halde $f(x) = 0$ denkleminin sıfıryerlerini belirleme problemini bu bölümün ilerleyen kısımlarında sayısal analiz sürecine örnekleri olarak ve ayrıca 6. Bölümde ise daha kapsamlı olarak inceliyoruz.

*İkinci problem*e baktığımızda $x_0 \neq x_1$ için (x_0, y_0),(x_1, y_1) noktalarından geçen doğru denkleminin

$$y = y_0 + \frac{y_1 - y_0}{x_1 - x_0}(x - x_0)$$

bağıntısıyla verildiğini hatırlayalım. Bu bağıntı yardımıyla

$$r \in (\min(x_i), \max(x_i)), i = 0, 1$$

noktasındaki bilinmeyen bir değer tahmin edilebilir. Ancak en genel halde, verilen keyfi sayıdaki(örneğin $n \geq 3$)

$$(x_0, y_0), (x_1, y_1), ..., (x_n, y_n)$$

nokta kümesini esas alarak, istenilen bir

$$r \in (\min(x_i), \max(x_i)), r \neq x_i, i = 0, 1, ..., n$$

[2] *Ömer Hayyam(1048-1131, İranlı bilim adamı)*

[3] *Girolamo Cardano (1501-1578, İtalyan bilim adamı)*

[4] Abel, Niels Henrik(1802-1829, Norveçli bilim adamı).

[5] Galois, Evariste(1811-1832). Galois gruplar teorisini genç yaşta geliştiren ve genç yaşta yaşamını yitiren Fransız matematikçi.

noktasındaki değeri tahmin etme işlemi olarak bilinen *interpolasyon* işlemi analitik olarak değerlendirilemez, yani çözüm için analitik yöntem mevcut olsa bile yöntem kağıt üzerinde ve kalem yardımıyla uygulanarak istenilen çözüm elde edilemez. Söz konusu problemi 4. Bölümde inceliyoruz.

Benzer biçimde verilen noktalara yakın noktalardan geçen en genel biçimdeki uygun bir eğrinin belirlenmesi problemi(*eğri uydurma*) analitik olarak incelenebilecek problem değildir, ve bu problemi 5. Bölümde inceliyoruz.

Üçüncü problem lineer bir denklem sistemidir ve çözümü yoketme yöntemi adını verilen bir analitik yöntemle kolayca belirlenebilir. Ancak denklem ve bilinmeyen sayısı artması durumunda ilgili sistemin "*kağıt-kalem*" çözümü aşırı dikkat ve zaman gerektirir ve hatta belirli bir noktadan sonra ise mümkün olmaz. Oysa özellikle mühendislikte bir çok araştırma problemi, binler, yüzbinler ve hatta milyonlarca bilinmeyenli lineer sistemlerin çözümünü gerektirmektedir. 7. Bölümde lineer denklem sistemleri için sıkça kullanılan sayısal yöntemleri inceliyoruz.

*Dördüncü problem*e gelince

$$\int_0^1 \sin(x)dx = 1 - \cos(1)$$

olduğunu kolayca görebiliriz, çünkü integrand olarak tanımlanan $f(x) = sin(x)$ fonksiyonunun belirsiz integrali mevcuttur ve *Kalkülüsün Temel Teoremi* yardımıyla istenilen sonucu kolayca elde edebiliriz. Ancak şimdi de

$$\int_0^1 \sin(x^2)dx$$

intregralini göz önüne alalım. Türevi $\sin(x^2)$ fonksiyonuna eşit olan ve sonlu sayıda elemanter fonksiyonunun lineer bileşimi biçiminde ifade edilebilen hiç bir fonksiyon bulamayız. Bu ve benzeri bir çok fonksiyonun sonlu sayıda elemanter fonksiyonun lineer bileşimi biçiminde ifade edilebilen belirsiz integrali mevcut değildir, yani belirsiz integralin elde edilmesi için analitik yöntem mevcut değildir. O halde bu problem için de analitik yöntemi uygun bir yöntem değildir. 8. Bölümde sıkça kullanılan sayısal integrasyon yöntemlerini inceliyoruz.

Son olarak *beşinci problem* birinci mertebeden sabit katsayılı bir başlangıç değer problemidir ve çözümü

$$y = t - 1 + (1 - a + y_0)e^{(a-t)}$$

olarak elde edilir. Ancak şimdi de

$$\begin{aligned} y\prime &= t - y^2, t \in (a, b) \\ y(a) &= y_0 \end{aligned}$$

başlangıç değer problemini göz önüne alalım. Bilinen analitik yöntemlerle problemi çözemeyiz? Analitik yöntemler çok özel denklem türleri(lineer, değişkenlerine ayrılabilen, homojen, vb) için geçerlidirler ve en genel bir Bayağı Diferensiyel Denklem için uygun değildirler. Benzer durum Kısmi Diferensiyel Denklemler için de geçerlidir. Bu amaçla geliştirilen ve *sonlu fark yöntemleri* adı verilen özel yöntemler sınıfını, "*Diferensiyel Denklemler için Sonlu Fark Yöntemleri[4]*" isimli çalışmada inceliyoruz.

Yukarıda gözönüne aldığımız problemlerin bir kısmında veri sayısının artması ve diğer bir kısmında ise problem formülasyonunda yapılan ufak değişiklikten sonra analitik yöntemlerin artık mümkün olmadığını gözlemliyoruz. Bu durumda sıkça başvurulan yöntemler *sayısal analiz* yöntemleridir.

Sayısal analiz, genellikle analitik yöntemle analizin mümkün olmadığı veya bu yöntemle elde edilen çözümün kullanışlı olmadığı durumlarda bilgisayar yardımıyla problemin gerçek ya da genellikle yaklaşık çözümünü belirleme yöntemidir.

Genellikle bilgisayar ortamında gerçekleştirilebilen sayısal analiz yöntemleri, matematiğin doğuşu itibariyle kullanılmakta olan analitik yöntemlere kıyasla oldukça yeni bir yöntem türüdür ve özellikle $XX.$ yüzyılın ikinci yarısında gelişen ve yaygınlaşan bilgisayar teknolojisine paralel olarak önem kazanmıştır. Sayısal yöntemler olarak adlandırılan sayısal analiz yöntemleri ise çok çeşitli araştırma alanlarında kullanılmakta olup, bu kaynağın esas konusunu teşkil etmektedirler.

Öte yandan matematiksel literatürde *kalitatif analiz* ise çözümü belirlemeksizin(ya da belirleyemeksizin) çözüm hakkında bilgi edinmeyi sağlayan analiz yöntemidir. Bu yöntem de modern ve kısmen yeni bir analiz yöntemidir. Örneğin bir diferensiyel denklemi çözmeden yön alanları yardımıyla çözüm eğrilerinin davranışı belirlenebilir. Denge çözümler ve bu çözümlerin kararlılık analizleri ilgili problemlerin çözümünü belirlemeden gerçekleştirilebilir.

Son olarak *sembolik analiz* yönteminden bahsedelim. Sembolik analiz, analitik analizi mümkün olan ancak daha çok işlem karmaşıklığına neden

olan problemlerin analitik çözümlerinin bilgisayar cebir programı adı verilen programlar yardımıyla elde etme yöntemidir. Sembolik analiz, *Maxima* [[6],[3]], veya *MATLAB* [[5],[2]] sembolik araç kutusu veya *Maple*, *Mathematica*, *Macysma* ve benzeri yazılımlar yardımıyla gerçekleştirilebilen ve kısmen yeni bir analiz yöntemidir.

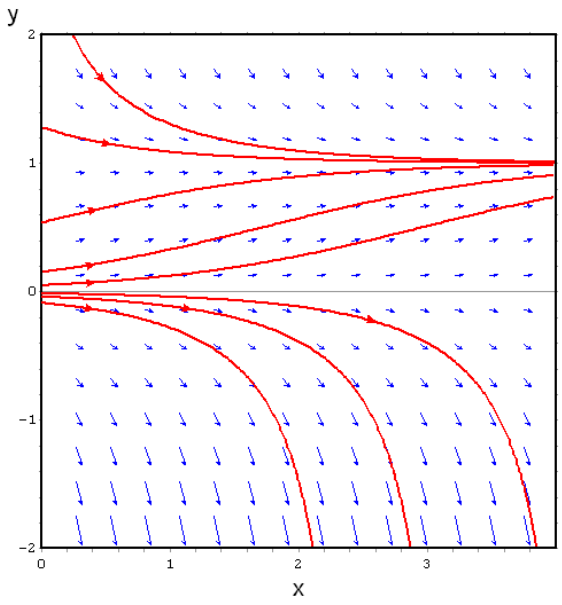
Şekil 1.1 de

$$y' = y(1-y)$$

diferensiyel denklemi için elde edilen yön alanları(kısa çizgiler) ve bazı çözüm eğrileri(sürekli çizgiler) görülmektedir. Şekil 1.1 de verilen yön alanları *Maxima* yazılımının *plotdf* fonksiyonu yardımıyla elde edilmiştir:

$$plotdf(y*(1-y),[x,0,4],[y,-2,2]);$$

Bu örnekte gösterilen yön alanları herhangi bir *yazılıma ihtiyaç duyulmadan da* elde edilebilir. Elde edilen yön alanlarına teğet olan çözüm eğrilerinin davranışı kolayca tahmin edilebilir.



Şekil 1.1: Kalitatif analiz örneği: $y\prime = y(1-y)$ denkleminin yön alanları ve bazı çözüm eğrileri.

Aşağıda verilen başlangıç değer problemini gözönüne alalım.

$$\begin{aligned} y'' + y' &= x \\ y(0) &= 0, y'(0) = 0 \end{aligned}$$

Verilen problemin analitik çözümünün

$$y = (x^2 - 2x + 2)/2 - e^{-x}$$

ile verildiği yukarıda bahsedilen yazılımlardan herhangi birisi yardımıyla kolayca görülebilir. Örneğin *Maxima* ortamında sözkonusu problemin çözümünü veren komutlar aşağıda sunulmaktadır:

```
(%i1)  denklem:'diff(y,x,2)+'diff(y,x)=x;
(denklem)  d^2/dx^2 y + d/dx y = x

(%i2)  cozum:ode2(denklem,y,x);
(cozum)  y = %k2 %e^-x + (x^2 - 2 x + 2)/2 + %k1

(%i3)  ic2(cozum,x=0,y=0,'diff(y,x)=0);
(%o3)  y = (x^2 - 2 x + 2)/2 - %e^-x
```

MATLAB/OCTAVE ve *Maxima*'nın matematiksel amaçla kullanımını, özetle ve sırasıyla [2] ve [3] nolu kaynaklarda inceliyoruz.

1.2 Sayısal analiz süreci

Sayısal analiz süreci

- uygun bir matematiksel dille ifade edilmiş bir problem ile başlayıp,
- problemin çözümü için gerekli sayısal yöntem(ve mümkünse yakınsaklık analizi),
- söz konusu sayısal yöntemin uygulanabilmesi için, takip edilmesi gereken adımların kümesi olarak adlandırılan algoritma[6],

[6] Algoritma sözcüğü El-Harizmi nin isminin latince okunuşundan türetilmiştir.[8]

- algoritmanın uygun bir programlama diline dönüştürülmüş kodu veya diğer deyimle programı,
- geliştirilen programın örnek problemler üzerinde test edilmesi(uygulama) ve
- sonuç, yorum, yöntemin kritiği ile mümkünse zayıf yönler için alternatif arayış aşamalarından oluşmaktadır.

Sayısal analiz aşamalarını öncelikle verilen bir nokta komşuluğunda sıfıryerini içeren aralığı belirleme problemi üzerinde inceleyelim:

1.2.1 Örnek 1: Verilen bir fonksiyonun, verilen bir nokta komşuluğunda reel sıfıryerini(eğer mevcutsa) içeren aralığı belirleme problemi

Verilen bir f fonksiyonunun ve yine verilen bir x_0 noktası komşuluğundaki sıfıryerini içeren $[a, b]$ aralığını belirleme problemini gözönüne alalım.

1. **Problem(Sıfıryerini içeren aralık belirleme)** Reel sayıların uygun bir alt kümesi üzerinde tanımlı ve reel sıfıryeri olan sürekli bir f fonksiyonun verilen bir x_0 noktasının $(x_0 - R, x_0 + R)$,örneğin $R = 10$,komşuluğundaki sıfıryerini içeren $[a, b]$ aralığını belirleyiniz.

2. **Sayısal yöntem(**sağ veya sol yönde tarama**):** x_0 noktasını içeren uygun bir $[x_{\min} = x_0 - R, x_{\max} = x_0 + R]$ kümesine sıfıryeri tarama aralığı adı verelim. $x = x_0$ noktasından başlayarak önce sağa doğru, uygun bir $h > 0$ adım uzunluğu ile ilerleyelim: $x < x_{\max}$ olduğu sürece

   $$f(x)f(x + h) <= 0$$

   eşitsizliğini sağlayan ilk $(x, x + h)$ nokta çiftini belirleyelim. Bu durumda $a = x, b = x + h$ dir.

   Eğer belirtilen kriterleri sağlayan nokta çifti bulunamaz ise, bu durumda $x = x_0$ noktasından başlayarak, $x > x_{\min}$ olduğu sürece

   $$f(x - h)f(x) \leq 0$$

   eşitsizliğini sağlayan ilk $(x - h, x)$ nokta çiftini belirleyelim. Bu durumda $a = x - h, b = x$ dir.

Eğer sol yönde tarama işleminde de belirtilen kriteri sağlayan nokta çifti bulunamaz ise bu durumda $[x_{\min}, x_{\max}]$ tarama aralığında sıfıryerini içeren alt aralık belirlenememiş olur.

3. **Algoritma** Yönteme ait Algoritma 1.1 aşağıda verilmektedir:

Algoritma 1.1 Sıfır yerini içeren aralık belirleme algoritması

(a) Girdi : f, x_0

(b) Varsayılan parametreler:

i. $R = 10$ varsayılan tarama yarıçapı

ii. $x_{\min} = x_0 - R, x_{\max} = x_0 + R$ sıfır yeri tarama aralığıdır.

iii. $h = 0.1$ ardışık noktalar arası mesafe, $x = x_0$ ilk tahmini değer

(c) Sağ yönde tarama:

i. $x < x_{\max}$ olduğu sürece

A. eğer $f(x)f(x+h) \leq 0$ ise $a = x, b = x + h$ tanımla ve çık

B. değilse $x = x + h$ olarak tanımla ve c(ii) ye git

(d) Sol yönde tarama

i. $x = x_0$

ii. $x > x_{\min}$ olduğu sürece

A. eğer $f(x-h)f(x) \leq 0$ ise $a = x - h, b = x$ tanımla ve çık

B. değilse $x = x - h$ olarak tanımla ve d(ii) ye git

(e) Sıfır yeri için tahmini aralık bulunadı yaz ve çık.

4. **Program(Kod)** Algoritma 1.1 e ait Program 1.1 aşağıda verilmektedir.

5. **Uygulama**

ÖRNEK 1.1. *$f(x) = exp(x) - x - 4$ fonksiyonunun $x_0 = 0$ noktası komşuluğundaki sıfıryerini içeren ve $h = 0.1$ uzunluklu $[a, b]$ aralığını belirleyiniz.*

```
%-------------------------------------------------------------
function [a,b]=bul(f,x0)
 x=x0; %  başlangıç noktası
         R=10;    % varsayılan tarama bölgesi yarıcapı
 xmin=x0-R; xmax=x0+R; % aralık uç noktaları
 h=0.1;   %tarama adım uzunluğu
 while x<xmax  %sağ yönde arama
   if f(x)*f(x+h)<=0 a=x;b=x+h; return;
   else
     x=x+h;
   end;
 end

   x=x0;
  while x>xmin    %sol yönde arama
   if f(x-h)*f(x)<=0 a=x-h,b=x; return;
   else
    x=x-h;
   end;
  end
 disp('sifiryerini içeren aralik bulunamadi');a=[];b=[];
%-------------------------------------------------------------
```

Program 1.1: Verilen nokta komşuluğunda sıfır yerini içeren aralık belirleme uygulaması

Çözüm.

```
>> f=inline('exp(x)-x-4')

f =

Inline function:

f(x) = exp(x)-x-4

>> X=bul(f,0)

X= 1.7000  1.8000
```

ÖRNEK 1.2. *$f(x) = log(x) - x + 4$ fonksiyonunun $x_0 = 10$ noktası komşuluğundaki sıfıryerini içeren ve $h = 0.1$ uzunluklu $[a, b]$ aralığını belirleyiniz.*

Çözüm.

```
>> f=inline('log(x)-x+4')

f =

Inline function:

f(x) = log(x)-x+4

>> X=bul(f,10)

X=5.7000  5.8000
```

6. **Yöntemin analizi ve geliştirme önerileri**

   - Süreksiz fonksiyonlar için süreksizlik noktalarını içeren aralık yukarıda tanımlanan yöntem ile yanlışlıkla sıfıryeri olarak yorumlanabilir. Örneğin $f(x) = 1/x$ fonsiyonuna sıfır noktasını içeren bir aralıkta yöntem uygulandığı taktirde bu tür bir yanlış sonuç oluşabilir. Yöntem sürekli fonksiyonlar için aradeğer teoremini esas aldığı için sadece sürekli fonksiyonlara uygulanabilir.

- Yöntem sürekli fonksiyonlar için aradeğer teoremini esas almaktadır ve sadece sıfır noktası komşuluğunda işaret değiştiren sıfır yerlerini(basit yani tek katlı sıfıryerlerini) belirlemek amacıyla kullanılabilir, fakat $f(x) = x^2$ gibi çift katlı sıfıryerlerine sahip olan, yani sıfıryeri komşuluğunda işaret değiştirmeyen fonksiyonların sıfıryerlerinin belirlenmesinde kullanılamaz.

Örnek 2 de ise verilen bir aralığın uç noktalarında işaret değiştiren fonksiyonun sıfıryerini belirleme problemi incelenmektedir.

1.2.2 Örnek 2: Verilen bir fonksiyonun, verilen bir aralıktaki sıfıryeri için yaklaşım belirleme problemi

1. **Problem:**

   f fonksiyonu $[a, b]$ aralığında tanımlı ve aralığının uç noktalarında işaret değiştiren (yani, $f(a)f(b) < 0$) sürekli bir fonksiyon olsun. Fonksiyonunun $[a, b]$ aralığındaki r sıfıryeri için yeterince küçük $\epsilon > 0$ ile $|f(c)| < \epsilon$ kriterini sağlayan $c \cong r$ yaklaşımını belirleyiniz.

   Sözkonusu problemin çözümü mevcuttur. Aşağıda ifade edilen ve Sürekli Fonksiyonlar için Aradeğer Teoremi(Teorem 1.1) olarak bilinen teorem gereğince f fonksiyonunun $[a, b]$ aralığında en az bir reel sıfıryeri olduğunu biliyoruz. O halde *$|f(c)| < \epsilon$ kriterini sağlayan en az bir $c \cong r$* sıfıryeri yaklaşımı mevcuttur. Öncelikle sürekli fonksiyonlar için aradeğer teoremini hatırlayalım:

   TEOREM 1.1. *(Sürekli fonksiyonlar için aradeğer teoremi) f fonksiyonu $[a, b]$ kapalı aralığında sürekli ve*

   $$m := \min_{a \le x \le b} f(x); M := \max_{a \le x \le b} f(x)$$

   olarak tanımlansın. Bu taktirde $\forall\ s \in [m, M]$ için $\exists r \in [a, b]$ öyle ki $f(r) = s$ dir.

   Basit bir biçimde ifade etmek gerekirse, sürekli fonksiyonlar için aradeğer teoremi, sürekli bir fonksiyonun kapalı aralıktaki en küçük ve en büyük değeri arasında kalan her değeri en az bir noktada alacağını ifade etmektedir.

Sonuç 1.1. *Aradeğer teoreminin bir sonucu olarak, eğer f fonksiyonu $[a,b]$ aralığında sürekli ve $f(a)f(b)<0$ ise $\exists r\in(a,b)$ öyle ki $f(r)=0$ dır.*

Çünkü $f(a)f(b)<0$ olması durumunda sıfır noktası $f(a)$ ile $f(b)$ arasındadır, o halde f fonksiyonu altında sıfır noktasına resmedilen $[a,b]$ aralığında en az bir nokta mevcut olmalıdır. Bu sonuca göre kapalı aralıkta sürekli olan ve bu aralığın uç noktalarında işaret değiştiren fonksiyonun bu aralık içerisinde en az bir sıfıryeri mevcuttur.

Bir sonraki aşama ise sözkonusu sıfıryerini elektronik ortamda belirleyecek olan ve *sayısal yöntem* olarak bilinen uygun bir matematiksel yöntemin belirlenmesidir.

2. **Sayısal Yöntem**(*ikiye bölme yöntemi*): Sayısal yöntem problemin elektronik ortamda çözümü için ne yapılması gerektiğini ifade eder.

   Bu amaçla ikiye bölme yöntemi adı verilen sayısal yöntemi kullanalım.

   Bu yöntem ile $[a,b]$ aralığı ile başlanarak, bu aralık her adımda iki eşit parçaya bölünür ve fonksiyonun işaret değiştirdiği alt aralık belirlenir. Aralık bölme işlemine fonksiyonun işaret değiştirdiği yeni aralık ile devam edilir. İşlemin ne zaman sonlandırılacağını *sonuçlandırma kriteri* adı verilen kriter belirler.

   Sonuçlandırma kriteri olarak, belirlenen yeni aralığın orta noktasındaki fonksiyon değerinin mutlak değerce verilen(veya tanımlanan) yeterince küçük bir $\epsilon>0$ dan küçük olma kriteri olarak kabul edelim. Bu durumda elde edilen en son alt aralığın orta noktasını sıfıryeri için bir yaklaşım olarak kabul edebiliriz.

   Öteyandan elde edilen en son aralığın uzunluğunun yeterince küçük pozitif bir sayıdan küçük olması kriteri gibi daha farklı sonuçlandırma kriteri de kullanılabilir.

ÖRNEK 1.3. *$f(x)=x^2-1$ fonksiyonunun $[0,3]$ aralığındaki sıfıryeri için ikiyebölme yöntemi ile ilk üç yaklaşımı elde ediniz.*

Çözüm.

f fonksiyonu sürekli ve

$$f(0)f(3)=-8<0$$

olduğu için fonksiyonun verilen aralıkta bir sıfıryeri mevcuttur ve ikiye bölme yöntemi uygulanabilir. Sıfıryerini içeren ve $n-$inci adımda($n \geq 1$) elde edilen aralığı $[a_n, b_n]$ ve bu aralığın orta noktasını da c_n ile gösterelim. Buna göre ilk adımda

$$a_1 = 0, b_1 = 3, c_1 = 3/2$$

elde ederiz.

$f(c_1) = 5/4 \neq 0$ olduğundan sıfıryeri henüz elde edilmemiştir, sıfıryerini içeren ikinci alt aralığı belirleyelim: Bu alt aralık ya

$$[a_1, c_1] = [0, 3/2] \text{ veya } [c_1, b_1] = [3/2, 3]$$

aralığı olmalıdır.

$$f(0)f(3/2) = -5/4 < 0$$

olduğu için yeni alt aralık

$$[a_2, b_2] = [0, 3/2] \text{ olup, bu aralığın orta noktası ise } c_2 = 3/4$$

tür.

$f(c_2) = f(3/4) = -7/16 \neq 0$ olduğundan sıfıryeri henüz elde edilmemiştir. Sıfıryerini içeren üçüncü alt aralığı da benzer biçimde belirleyebiliriz: Bu alt aralık ya

$$[a_2, c_2] = [0, 3/4] \text{ veya } [c_2, b_2] = [3/4, 3/2]$$

aralığı olmalıdır.

$$f(3/4)f(3/2) = -35/64 < 0$$

olduğu için yeni alt aralık

$$[a_3, b_3] = [3/4, 3/2] \text{ ve bu aralığın orta noktası } c_3 = 9/8$$

dir. Dolayısıyla verilen fonksiyonun sıfıryeri için problemde istenilen ilk üç yaklaşımı

$$c_1 = 3/2, c_2 = 3/4, c_3 = 9/8$$

olarak elde etmiş olduk.

Aşağıda göreceğimiz üzere ikiye bölme yöntemi ile her bir adımda elde edilen aralığın orta noktası olarak tanımlanan $\{c_n\}$ dizisi fonksiyonun sıfıryerine yakınsamaktadır.

Ancak verilen keyfi bir fonksiyonun sıfıryerini belirlemek için gerekli işlemleri yukarıda olduğu gibi kağıt ve kalem ile gerçekleştiremeyiz. Bilgisayar ortamında gerçekleştirilmesi gereken bu işlemler için her bir adımda gerçekleştirilmesi gerekli olan işlemler açık ve net olarak ifade edilmelidir.

3. **Algoritma**: Sayısal yöntemin *hangi adımlar* takip edilerek ve *nasıl* uygulanacağını ifade eder. Daha açık bir ifade ile Algoritma

   - *input* adı verilen verilerin alınması,
   - yöntemin icrası için gerekli her bir adım ve
   - kullanıcıya iletilecek sonuçların(*Çıktı* veya *Output*) açık ve net bir biçimde ifade edildiği **komutlar kümesidir**.

   Model problemimiz için *Algoritma* 1.2 aşağıda sunulmaktadır. Bu algoritmada her bir adımda elde edilen alt aralık yine $[a, b]$ aralığı olarak tanımlanmaktadır, Örnek 1 in aksine burada indis kullanmıyoruz.

Algoritma 1.2 ikiye bolme yontemi algoritması

(a) Girdi: f, a, b, ϵ

(b) Eğer $f(a)f(b) > 0$ ise cık

(c) $c = (a+b)/2$ alt aralığın orta noktası olmak üzere $a, c, b, f(c)$ degerlerini yaz

(d) $|f(c)| > \epsilon$ olduğu sürece i,ii,iii adımlarını tekrarla

   i. Yeni alt aralığı belirle: eğer $f(a)f(c) < 0$ ise $b = c$, değilse $a = c$ (sıfır yerini içeren yeni $[a, b]$ aralığı)

   ii. Yeni alt aralığın orta noktası: $c = (a + b)/2$

   iii. $a, c, b, f(c)$ degerlerini yaz

(a) adımında kullanıcının sıfıryerini belirlemek istediği fonksiyonu ve bu fonksiyonun işaret değiştirdiği aralığının a ve b ile gösterilen uç noktalarını tanımlaması istenmektedir. Ayrıca sıfıryeri için uygun bir yaklaşımın belirlendiğini test yapmak amacıyla yeterince küçük $\epsilon > 0$ sabitinin tanımlanması istenmektedir.

(b) de $f(a)f(b) > 0$ olması durumda yöntemin çalışmayacağı ifade edilmektedir.

(c) de $[a, b]$ aralığının orta noktası belirlenerek c değişkenine atanmakta ve $a, c, b, f(c)$ değerleri yazdırılmaktadır.

(d) de $|f(c)| > \epsilon$ olduğu sürece

$(d) - (i)$ de sıfır yerini içeren alt aralık belirlenerek, bu alt aralık tekrar $[a, b]$ aralığı olarak adlandırılmaktadır. Eğer $f(a)f(c) < 0$ ise yeni $[a, b] = [a, c]$, yani $b = c$, değilse yeni $[a, b] = [c, b]$, yani $a = c$ dir.

$(d) - (ii)$ de yeni alt aralığın orta noktası belirlenmekte

$(d) - (iii)$ de ise $a, c, b, f(c)$ değerleri yazdırılmaktadır.

4. **Program(Kod)**: Bir sonraki aşama ise algoritması belirlenen probleme ait program geliştirme aşamasıdır. Bu aşama algoritma ile belirlenen komutlar kümesinin Bilgisayar Dili'ne dönüştürülmesi aşamasıdır. Bu bağlamda *Programlama Dili* olarak adlandırılan uygun bir dil(*Basic, Pascal, Fortran, C*, vs) veya *Programlanabilme özelliğine sahip yazılım* kullanılır.

   Uygulamalarımız için çoğunlukla *MATLAB* veya aynı yazım kurallarını kullanan *OCTAVE*'ı [2] kullanıyoruz. Yukarıdaki problem için *OCTAVE* programı aşağıda verilmektedir.

   OCTAVE $(http : //www.gnu.org/software/OCTAVE)$ adresinden kolayca erişilebilen ücretsiz bir yazılımdır ve matematiksel işlemler için kullanımı [2] nolu kaynakta özet olarak incelenmektedir.

5. **Uygulama**

   ÖRNEK 1.4. *$f(x) = cos(x) - x$ fonksiyonunun $[0, 2]$ aralığındaki sıfıryerini ikiye bölme yöntemi yardımıyla belirleyiniz.*

   Çözüm.

   f fonksiyonunu

   ```
   f=inline('cos(x)-x');
   ```

   komutu ile tanımlayalım. Daha sonra aşağıdaki komut yardımıyla Program 1.2 i çalıştırarak aşağıda gösterilen sonuçları elde edebiliriz.

```
%-------------------------------------------------------

function c=ikibol(f,a,b)
% f fonksiyonunun [a,b] aralığındaki
% sıfır yerini ikiye bölme yöntemi ile bulur.
% Yazılımı: c=ikibol(f,a,b)

        eps=1e-5; %parametre
if f(a)*f(b)>0
   error("ikiye bolme yöntemi uygulanamaz");
end;

c=(a+b)/2;
fprintf('    a          c          b              f(c)  \n');
fprintf('%9.6f  %9.6f  %9.6f  %9.6f\n',a,c,b,f(c));
while  abs(f(c))>eps
if f(a)*f(c)< 0
   b=c;
else
  a=c;
end
  c=(a+b)/2;
 fprintf('%9.6f  %9.6f  %9.6f  %9.6f  \n',a,c,b,f(c));
 end

%-------------------------------------------------------
```

Program 1.2: Matlab veya Octave ile ikiye bölme yöntemi uygulaması

```
>>ikibol(f,0,2)

 0.000000  1.000000  2.000000 -0.459698
 0.000000  0.500000  1.000000  0.377583
 0.500000  0.750000  1.000000 -0.018311
 0.500000  0.625000  0.750000  0.185963
 0.625000  0.687500  0.750000  0.085335
 0.687500  0.718750  0.750000  0.033879
 0.718750  0.734375  0.750000  0.007875
 0.734375  0.742188  0.750000 -0.005196
 0.734375  0.738281  0.742188  0.001345
 0.738281  0.740234  0.742188 -0.001924
 0.738281  0.739258  0.740234 -0.000289
 0.738281  0.738770  0.739258  0.000528
 0.738770  0.739014  0.739258  0.000120
 0.739014  0.739136  0.739258 -0.000085
 0.739014  0.739075  0.739136  0.000017
 0.739075  0.739105  0.739136 -0.000034
 0.739075  0.739090  0.739105 -0.000008

ans =

    0.7391
```

6. **Yöntemin Analizi**

   Hazırlanan programın doğru çalıştığının kontrol edilmesini takip eden son aşama ise elde edilen sonuçların irdelenmesi ve yorumlanmasıdır. Bu bağlamda cevaplandırılması gereken sorular:

   - Yöntemin söz konusu aralıktaki sıfıryerini her zaman belirleyip ya da belirleyemeyeceği(Teorem 1.2) ve
   - Sıfıryerini belirleyebilme hızı(orta noktalardan oluşan dizinin yakınsama hızıdır.

   Bu sorulara vereceğimiz cevaplar yardımıyla *yöntemin yakınsaklığı, yakınsama hızı* ve *bilgisayar sistem kaynaklarını hangi düzeyde kullandığı* konusunda bilgi sahibi oluruz. Bu bilgiler *yöntemin **"kimlik"** bilgileridir* ve bir diğer yöntemle karşılaştırılırken kullanılır.

TEOREM 1.2. *f fonksiyonu $[a,b]=[a_1,b_1]$ aralığının uç noktalarında işaret değiştiren sürekli bir fonksiyon ve r de fonksiyonun her n için $f(a_n)f(b_n)<0$ şartını sağlayan $[a_n,b_n]$ aralığındaki sıfıryeri ve $\{c_n\}_{n=1}^{\infty}$ ise $c_n=(a_n+b_n)/2$ ile tanımlanan orta noktalar dizisi olsun. Bu taktirde*

$$\lim_{n\to\infty} c_n = r$$

dir.

r sıfıryeri için

$$|r-c_n| \le \frac{b_n-a_n}{2}$$

olduğu açıktır. Öte yandan her bir alt aralığın uzunluğu önceki alt aralığın uzunluğunun yarısına eşit olduğundan

$$|r-c_n| \le \frac{b_n-a_n}{2} = \frac{b_{n-1}-a_{n-1}}{2^2} = \cdots = \frac{b_1-a_1}{2^n}$$

elde ederiz. Yukarıdaki eşitsizlikten

$$\lim_{n\to\infty} |r-c_n| = 0$$

elde ederiz. Öte yandan

$$-|r-c_n| \le r-c_n \le |r-c_n|$$

eşitsizliği ve genel matematik dersinden bilinen *sıkıştırma teoreminden [9]* sonuç açıkça görülür.

Şimdi de yakınsama hızını belirlemek amacıyla "yakınsama basamağı" kavramını tanımlayalım.

TANIM 1.1. *Bir r noktasına yakınsayan $\{c_n\}_{n=0}^{\infty}$ dizisi verilmiş olsun. Eğer her $n \ge N$ için*

$$|r-c_{n+1}| \le \alpha |r-c_n|^{\beta}$$

eşitsizliği sağlanacak biçimde $\alpha>0, \beta \ge 1$ reel sayıları ve $N>0$ tamsayısı mevcutsa $\{c_n\}_{n=1}^{\infty}$ dizisi β-ıncı basamaktan yakınsak bir dizidir denir. $\beta=1$ olması durumunda yakınsama için $\alpha \in (0,1)$ olmalıdır ve bu durumda diziye lineer yakınsak dizi adı verilir. $\beta=2$ olması durumda ise diziye kuadratik yakınsak dizi adı verilir.

- İkiye bölme yöntemi için

$$|r - c_{n+1}| \leq \frac{b_{n+1} - a_{n+1}}{2} = \frac{1}{2}\frac{b_n - a_n}{2}$$

ve

$$|r - c_n| \leq \frac{b_n - a_n}{2}$$

eşitsizliklerini karşılaştırarak

$$|r - c_{n+1}| \cong \frac{1}{2}|r - c_n|$$

elde ederiz. O halde yöntem lineer olarak yakınsaktır ve

$$|r - c_{n+1}|/|r - c_n| \cong \frac{1}{2}$$

sabiti ise *ortalama yakınsaklık oranı* olarak tanımlanır.

7. **Geliştirme önerileri ve alternatif yöntemler**

En iyi sayısal yöntem, gerçek sonucu en büyük hassasiyetle ve minimal bilgisayar bellek ve zaman kaynağı kullanımı ile elde eden yöntemdir. Alternatif olarak daha değişik yöntemler uygulanabilir:

- **Kirişle Bölme(Regula Falsi, false position) yöntemi***:* Genellikle *Regula Falsi* olarak bilinen yöntem, her adımda aralığı orta noktasından ikiye bölmek yerine, $f(a)f(b) < 0$ olmak üzere $(a, f(a))$, $(b, f(b))$ noktalarını birleştiren kirişin x eksenini kesim noktası yardımıyla aralığı iki alt parçaya böler. Kirişin ekseni kesim noktasını belirlemek için öncelikle kiriş denklemini gözönüne alalım:

$$y = f(a) + \frac{f(b) - f(a)}{b - a}(x - a)$$

Bu doğrunun $x = c$ olarak adlandıracağımız x eksenini kesim noktasını, $y = 0$ alarak

$$\begin{aligned} x &= c = a - \frac{(b-a)}{f(b) - f(a)} f(a) \\ &= b - \frac{(b-a)}{f(b) - f(a)} f(b) \end{aligned} \tag{1.1}$$

olarak elde ederiz.

Bu yönteme ait algoritmayı, Algoritma 1.2 de küçük bir değişiklik yapmak suretiyle elde edebiliriz: Algoritma 1.2 de

$$c = (a + b)/2$$

yerine

$$c = b - \frac{(b-a)}{f(b) - f(a)} f(b)$$

almak yeterlidir.

ÖRNEK 1.5. *$f(x) = x^2 - 1$ fonksiyonunun $[0, 3]$ aralığındaki sıfıryeri için kirişle bölme yöntemi ile ilk üç yaklaşımı elde ediniz.*

Çözüm.

f fonksiyonu sürekli ve

$$f(0)f(3) = -8 < 0$$

olduğu için fonksiyonun verilen aralıkta bir sıfıryeri mevcuttur ve ikiye bölme yöntemi uygulanabilir. Sıfıryerini içeren ve $n-$inci adımda($n \geq 1$) elde edilen aralığı $[a_n, b_n]$ ve $(a_n, f(a_n))$,$(b_n, f(b_n))$ noktalarını birleştiren kirişin x eksenini kestiği noktayı da c_n ile gösterelim. Buna göre

$$c_n = b_n - \frac{(b_n - a_n)}{f(b_n) - f(a_n)} f(b_n)$$

dir. İlk adımda

$$a_1 = 0, b_1 = 3, c_1 = \frac{1}{3}$$

dir.

$f(1/3) = -8/9 \neq 0$ olup, sıfıryeri henüz belirlenmemiştir. Şimdi sıfıryerini içeren ikinci alt aralığı belirleyelim: Bu alt aralık ya

$$[a_1, c_1] = [0, 1/3] \text{ veya } [c_1, b_1] = [1/3, 3]$$

aralığı olmalıdır.

$$f(1/3)f(3) = -64/9 < 0$$

olduğu için yeni alt aralık

$$[a_2, b_2] = [1/3, 3]$$

ve

$$c_2 = 3/5$$

dir.

$f(3/5) = 9/25 - 1 = -16/25 \neq 0$ olup, sıfıryeri henüz belirlenmemiştir. Sıfıryerini içeren üçüncü alt aralığı da benzer biçimde belirleyabiliriz: Bu alt aralık ya

$$[a_2, c_2] = [1/3, 3/5] \text{ veya } [c_2, b_2] = [3/5, 3]$$

aralığı olmalıdır.

$$f(3/5)f(3) = -128/25 < 0$$

olduğu için yeni alt aralık

$$[a_3, b_3] = [3/5, 3]$$

ve

$$c_3 = 7/9$$

dur.

O halde sıfıryeri için kirişle bölme yöntemi ile elde ettiğimiz ilk üç yaklaşımı

$$c_1 = 1/3, c_2 = 3/5, c_3 = 7/9$$

olarak ifade edebiliriz.

Kirişle bölme yöntemi de her bir adımda elde edilen $[a_n, b_n]$ aralığında yer alan ve $(a_n, f(a_n)), (b_n, f(b_n))$ noktalarını birleştiren kirişin x eksenini kesim noktası olarak elde edilen $\{c_n\}$ noktalar dizisinin fonksiyonun sıfıryerine yakınsadığını öngörür(Alıştırma 15).

Uyarı. *ikibol.m isimli dosyada da*

$$c = b - \frac{(b-a)}{f(b) - f(a)} f(b)$$

değişikliğini yaparak elde ettiğiniz programı kirislebol.m *isimli dosyada kaydediniz.* function ikibol(f,a,b) *satırını ise* function kirislebol(f,a,b) *olarak değiştirmeyi unutmayınız(Alıştırma 6).*

Bir diğer alternatif ise *kirişlerle yaklaşım yöntemidir:*

- **Kirişlerle yaklaşım Yöntemi***:* Kirişle bölme yöntemindeki sıfıryerini içeren aralıkla başlayıp(yani $f(a)f(b) < 0$) ve yine sıfıryerini içeren alt aralık ile devam etme kriterinden vazgeçilebilir.
  - Bu durumda verilen f fonksiyonu ve belirlenmesi istenilen sıfıryerine yakın komşuluktaki keyfi iki x_1, x_2 noktası ile başlayarak $(x_1, f(x_1)), (x_2, f(x_2))$ noktalarından geçen kirişin x eksenini kestiği x_3 noktası (1.1)

$$x_3 = x_2 - \frac{(x_2 - x_1)}{f(x_2) - f(x_1)} f(x_2)$$

  olarak ta ifade edilebilir.
  - Daha sonra en güncel iki yaklaşım noktası olarak $x_1 := x_2$, $x_2 := x_3$ alarak, işleme $|f(x_3)| > \epsilon$ kriteri doğru olduğu sürece devam edilir. Elde edilen en son x_3 değeri sıfıryeri için yaklaşım kabul edilir. Bu yöntem kirişle bölme yöntemi ile çoğu kez karıştırılır ve kısaca *Kiriş(secant) veya Kirişlerle yaklaşım yöntemi* olarak adlandırılır(Bölüm 6).
- **Parabollerle yaklaşım yöntemi:** Kirişlerle yaklaşım yöntemi de geliştirilebilir.
- Sıfır yeri komşuluğunda iki nokta yerine, x_1, x_2 ve x_3 gibi üç nokta alarak,

$$(x_1,f(x_1)),(x_2,f(x_2)),(x_3,f(x_3))$$

  noktalarından geçen ikinci dereceden polinomun x_4 ile göstereceğimiz sıfır yerini fonksiyonun sıfır yeri için yaklaşım kabul edelim. Bir sonraki adımda $x_1 = x_2, x_2 = x_3, x_3 = x_4$ alarak işleme devam edelim. $|f(x_3)| > epsilon$ olduğu sürece işleme devam edelim. Kriteri sağlamayan ilk x_3 değerini sıfır yeri olarak kabul edelim. Yukarıda ana hatlarıyla bahsedilen ve parabollerin sıfır yerleri ile verilen fonksiyonun sıfır yerine yaklaşımı esas alan bu yöntem 1956 yıllında David E. Muller[7] tarafından geliştirilmiş olup Muller yöntemi olarak bilinir([1])(Bölüm 6).

[7]David E. Muller(1924-2008) Amerikalı matematik ve bilgisayar bilimci.

Örnek 3 ile verilen bir sonraki örneğimizde, sayısal analiz aşamalarını bir fonksiyonun verilen bir nokta komşuluğundaki **sıfıryerini içeren aralığı belirleme** problemi üzerinde inceliyoruz.

1.2.3 Örnek 3: Verilen bir fonksiyonun, verilen bir nokta komşuluğundaki sıfıryeri için yaklaşım belirleme problemi

Örnek 1 de geliştirilen yöntem ile Örnek 2 deki yöntemi birleştirerek karma(**hibrid**) olarak adlandırılan bir yöntem geliştirebiliriz.

1. **Problem**: Verilen bir f fonksiyonunun yine verilen bir x_0 noktası komşuluğundaki sıfıryerini belirleyiniz.
2. **Yöntem(karma yöntem):** Öncelikle verilen bir f fonksiyonunun yine verilen bir x_0 noktası komşuluğundaki sıfıryerini içeren $[a, b]$ aralığını Örnek 1 de geliştirdiğimiz yöntem ile belirledikten sonra, elde edilen aralığı ikiye bölme yöntemine göndererek fonsiyonunu sıfıryerini belirleyebiliriz.
3. **Algoritma** Yönteme ait algoritma aşağıda verilmektedir.

Algoritma 1.3 Verilen nokta komşuluğunda sıfıryeri belirleme algoritması.

(a) Girdi f,x0

(b) f nin sıfıryerini içeren [a,b] aralığını Örnek 1 deki yöntem ile belirle

(c) i. eğer f(a)=0 ise c=a,
ii. değil ve eğer f(b)=0 ise c=b dir,
iii. değilse Örnek 2 deki yöntem ile c=ikibol(f,a,b) ile c yi bul

(d) Çıktı: c

4. **Program** Algoritmaya ait program aşağıda verilmektedir.
5. **Uygulama**

ÖRNEK 1.6. *$f(x) = log(x) - x + 4$ fonksiyonunun $x_0 = 4$ noktası komşuluğundaki sıfıryerini belirleyiniz.*

```
%-------------------------------------------------------------
    function c=fsifir(f,x0);
    X=bul(f,x0);
    if isempty(X)
        c=[]; return;
      else
        a=X(1);b=X(2);
     if f(a)==0 c=a;
        elseif f(b)==0 c=b;
        else
         c=ikibol(f,a,b);
     end
   end
%-------------------------------------------------------------
```

Program 1.3: Girilen nokta komşulundaki sıfıryerini belirleme uygulaması.

Çözüm.

```
>> f=inline('log(x)-x+4')

f =

Inline function:

f(x) = log(x)-x+4

>> fsifir(f,4)

ans =

5.7490
```

Aynı işlem MATLAB/OCTAVE fzero fonksiyonu yardımıyla da gerçekleştirilebilir:

```
>> fzero(f,4)
```

```
ans =

5.7490
```

ÖRNEK 1.7. *$f(x) = xsin(1/x)$ fonksiyonunun $x_0 = 4$ noktası komşuluğundaki sıfıryerini belirleyiniz.*

Çözüm.

```
>> f=inline('x*sin(1/x)')

f =

Inline function:

f(x) = x*sin(1/x)

>> fsifir(f,4)

ans =

0.3183
```

elde ederiz.

Benzer yazılımlarla karşılaştırma:

MATLAB/OCTAVE *fzero* fonksiyonunun yukarıda tanımlanan f fonksiyonunun sıfıryerini $x_0 = 4$ başlangıç noktasıyla belirleyememektedir:

```
>> fzero(f,4)

Exiting fzero: aborting search for an interval containing
a sign change because NaN or Inf function value encountered
during search.

(Function value at -Inf is NaN.)
```

```
Check function or try again with a different starting value.
ans =NaN
```

Ancak başlangıç noktası fonksiyonun sıfıryerine daha yakın seçilerek, *fzero* yardımıyla da sıfıryeri belirlenebilmektedir:

>>fzero(f,1)

ans =

0.3183

6. **Yöntemin analizi ve Alternatif arayışlar**

Verilen bir reel x_0 noktası komşuluğunda basit ve reel sıfıryerini bulan bu yöntem

(a) çift katlı sıfıryerlerini ve

(b) karmaşık sıfıryerini belirleyemez.

Verilen bir reel x_0 noktası komşuluğunda reel sıfıryerini bulan yöntem, karmaşık sıfıryerini bulmak için geliştirilebilir.

Örnek 1-3 ile sayı sal analiz sürecine örnek olarak incelediğimiz ikiye bölme ve kirişle bölme yöntemleri **bölümleme yöntemleri** olarak adlandırılırlar, çünkü sıfıryerini içeren aralık, mevcut aralığın uygun bir biçimde bölün suretiyle elde edilmektedir.

Hatırlatma 1.1. *Yukarıda kısaca özetlenen kirişler ve parabolerle yaklaşım yöntemleri* ***yinelemeli yöntemler*** *olarak adlandırılırlar ve bu yöntemleri Bölüm* 6 *da inceliyoruz.*

Alıştırmalar 1.1.

1.

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, X = \begin{bmatrix} x \\ y \end{bmatrix}$$

olmak üzere $AX = b$ *denklem sisteminin çözümünü belirleme problemiinin sayısal analizini aşağıdaki adımları uygulayarak gerçekleştiriniz:*

(a) $det(A) \neq 0$ *ise Cramer yöntemini sayısal yöntem olarak deneyiniz.* $det(A) = 0$ *ise yöntemin uygulanamayacağı mesajını kullanıcıya iletiniz.*

(b) *Yönteme ait algoritmanızı adım adım ve açık bir biçimde yazınız. Algoritma kullanıcıdan* $A_{2\times 2}$ *matrisi ve* $b_{2\times 1}$ *vektörünü alarak* $X_{2\times 1}$ *çözümünü sunmalıdır.*

(c) *Algoritmaya ait programınızı OCTAVE yazılım diline uygun olarak geliştiriniz.*

(d) *Programınız farklı* A *matrisleri ve* b *sağ yan vektörleri için test yapınız.*

2.

$$ax^2 + bx + c = 0$$

denkleminin köklerini belirleme probleminin sayısal analizini aşağıdaki adımları uygulayarak gerçekleştiriniz.

(a) $a \neq 0$ *için bilinen kuadratik polinom kök formüllerini sayısal yöntem olarak deneyiniz.*

(b) *Yönteme ait algoritmanızı adım adım ve açık bir biçimde yazınız. Algoritma kullanıcıdan* a, b, c *katsayılarını alarak, reel veya kompleks kökleri belirlemelidir.*

(c) *Algoritmaya ait programınızı OCTAVE yazılım dilinde geliştiriniz.*

(d) *Programınızı farklı* a, b, c *katsayıları için test yapınız.*

(e) $a = 0$ *olması durumundaki kökü ayrıca hesaplatmayı unutmayınız.*

3. *Soru 2 deki analiziniz yardımıyla* $A_{2\times 2}$ *matrisinin özdeğerlerini belirleme probleminin sayısal analizini gerçekleştiriniz.*

4.

$$\begin{aligned} x^2 + y^2 &= r^2 \\ -ax^2 + y &= 0 \end{aligned}$$

çember ve parabolünün arakesit noktalarını belirleme probleminin sayısal analizini gerçekleştiriniz. Verilen r *yarıçapı ve* $a \neq 0$ *katsayısı için her iki arakesit noktası belirlenmelidir.*

5. $f(x) = x^2 - 5x$, $a = -2, b = 3$ *verilsin.* f *nin* $[a, b]$ *aralığındaki sıfıryeri için ilk üç yaklaşımı*

   (a) *ikiye bölme,*

   (b) *kirişle bölme yöntemleri ile belirleyiniz.*

6. $f(x) = x^3 - x - 1$, $a = -2, b = 3$ *verilsin.* f *nin* $[a, b]$ *aralığındaki sıfıryeri için ilk üç yaklaşımı*

   (a) *ikiye bölme,*

   (b) *kirişle bölme yöntemleri ile belirleyiniz.*

7. *İkiye bölme yöntemine ait Program 1.2 yi aşağıda verilen fonksiyonlar ve* $[a, b]$ *aralıkları için çalıştırarak sıfıryerlerini belirleyiniz. Epsilon sabitini* $eps = 1e - 10$ *olarak alınız.*

   (a) $f(x) = x^2 - 5x$, $a = -2, b = 3$

   (b) $f(x) = x^3 - x - 1$, $a = -2, b = 3$

   (c) $f(x) = ln(x + 1) - \frac{1}{4}x^2$, $a = 1, b = 4$

   (d) $f(x) = 5e^{-x} - coshx$, $a = 0, b = 4$

8. *Soru 7 yi kirişle bölme(regula falsi) yöntemi için tekrarlayınız.*

9. *Kullanıcıdan* f *fonksiyonu ,* a,b *değerleri ve* eps *toleransı ile birlikte yontem isimli bir değişken değerini de alarak* $yontem = 1$ *olması durumunda ikiye bölme yöntemini,* $yontem = 2$ *olması durumunda kirişle bölme yöntemini çağırarak sıfıryerini belirleyen bir uygulama geliştiriniz.*

10. *Soru 7 de verilen fonksiyonlar ve yine verilen* a *değerleri için* $x_0 = a$ *alarak Program 1.1 yardımıyla sıfıryerlerini içeren aralıkları hesaplayınız.*

11. *İkiye bölme yöntemini, her adımda elde edilen aralık içerisinde rasgele üretilen bir noktayı(örneğin MATLAB/OCTAVE ortamında* $rand$ *fonksiyonu ile)* c *noktası olarak kabul edecek biçimde modifiye ediniz. Elde ettiğiniz yöntemi ikiye bölme yöntemiyle karşılaştırınız. Ne gözlemliyorsunuz?*

12. *Program 1.3 ile tanımlanan $fsifir$ programı ve MATLAB/OCTAVE'a ait $fzero$ programları yardımıyla Soru 7 de verilen fonksiyonların sıfıryerlerini yine aynı soru da verilen a değerlerini başlangıç noktası alarak hesaplayınız.*

13. *MATLAB/OCTAVE ortamında $roots$ komutu, katsayıları verilen polinomun sıfır yerlerini belirler. Buna göre Soru 7(a),(b) deki polinomların katsayılarını girerek köklerini elde ediniz. Elde ettiğiniz sonuçlar yukarıdaki formül ile elde edilen sonuçlarla aynı olmalıdır. Örneğin*

    $>> roots([a\ b\ c])$ *komutu ile*

    $$p(x)=ax^2+bx+c$$

    polinomunun kökleri hesaplanır.

14. *Kirişle bölme yöntemi de her bir adımda elde edilen $[a_n, b_n]$ aralığında yer alan ve $(a_n, f(a_n)), (b_n, f(b_n))$ noktalarını birleştiren kirişin x eksenini kesim noktası olarak elde edilen $\{c_n\}$ noktalar dizisinin fonksiyonun sıfıryerine yakınsadığını gösteriniz.*

15. *Bilgisayarınızın bellek kullanım kapasitesini test yapınız:*

    MATLAB/OCTAVE ortamında $A = rand(n)$ komutu ile $n = 1000$ için rasgele bir $A_{n\times n}$ matrisini üreterek, matrisin tersi, determinantı ve rankını bulmaya çalışınız.

    $$n = 2000, 4000, 10000$$

    için aynı işlemleri yapmaya çalışarak tersini hesaplayabileceğiniz en büyük boyutlu matrisi tahmin etmeye çalışınız.

16. *Rasgele üreteceğiniz A matrisi ve* b *vektörü için MATLAB/OCTAVE ortamında $\mathbf{x} = A\backslash\mathbf{b}$ komutuyla çözebileceğiniz en büyük boyutlu $A\mathbf{x} = \mathbf{b}$ denklem sisteminin boyutunu belirlemeye çalışınız.*

17. *Bilgisayarınızın CPU hızını test yapınız:*

    $$S_N = \sum_{k=1}^{N} 1/k$$

    toplamını

    $$N = 1000, 10000, 100000, 1000000$$

değerleri için hesaplayarak her bir işlem için kullanılan CPU zamanını belirlemeye çalışınız. Artan N değerleri için CPU zamanı nasıl değişmektedir. N değerlerine karşı, toplama işlemi için gerekli CPU zamanlarının grafiğini çizdiriniz. Bunun için MATLAB/OCTAVE ortamında $cputime$ komutunu uygulayan aşağıdaki programı kullanabilirsiniz:

```
%-------------------------------------------------------
        function sonuc=topla(N)
        topla=0;
        zaman=cputime;
        for i=1:N
           topla=topla+1/i;
        end
        zaman=cputime-zaman;
        disp(['zaman=',num2str(zaman)]);
%-------------------------------------------------------
```

Program 1.4: For döngüsü ile seri toplamı için cpu testi.

Kaynaklar

[1] Atkinson, K. An Introduction to Numerical Analysis, John Wiley ve Sons, 1988.

[2] Coşkun, E. OCTAVE ile Sayısal Hesaplama ve Kodlama(URL:erhancoskun.com.tr).

[3] Coşkun, E. Maxima ile Sembolik Hesaplama ve Kodlama(URL:erhancoskun.com.tr).

[4] Coşkun, E. Diferensiyel Denklemler için Sonlu Fark Yöntemleri(URL:erhancoskun.com.tr).

[5] MATLAB, Mathworks(*URL:mathworks.com).*

[6] Maxima, GNU özgür yazılım(*URL:maxima.sourceforge.net*).

[7] OCTAVE, GNU özgür yazılım(*URL:OCTAVE.sourceforge.net*).

[8] Pottmeyer, L., News on quadratic polynomials, Snapshots of modern mathematics from Oberfolfach, 2/2017(URL:imaginary.org).

[9] S. W., Warren, Zill, D. G., Calculus: Early Transcendentals, Çeviri: Matematik Cilt I, II(Çeviri editörü İsmail Naci Cangül), Nobel Akademik Yayıncılık, 2010.

Bölüm 2

Bilgisayar Sayı Sistemi, Aritmetiği ve Hata

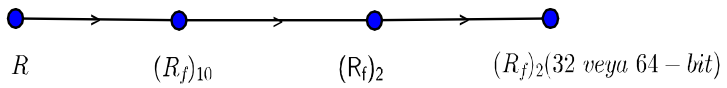
Sayısal yöntemler için gerekli en temel araçlar bilgisayarlardır, ancak bilgisayarlar reel veya kompleks sayılar kümesi yerine, bu kümelerin sonlu sayıda elemandan oluşan *çok özel* alt kümeleri üzerinde çalışırlar. Bu bölümde öncelikle *bilgisayar sayı sistemi* adı verilen bu *çok özel* alt kümeyi ve bu kümenin özelliklerini inceleyeceğiz. Ayrıca bilgisayar sayı sistemi üzerindeki aritmetik işlemleri inceleyerek, bu işlemlerin reel sayılar kümesi üzerinde alışık olduğumuz kapalılık ve birleşme gibi elemanter özelliklere sahip olmadığını göreceğiz. Böylece gerek bilgisayar sayı sistemi ve gerekse sayı sistemi üzerindeki aritmetik işlemleri yakından tanıyarak, sayısal analiz sürecinde sayı sistemi ve aritmetiğinden kaynaklanabilecek hataları minimize etmenin yollarını öğrenmiş olacağız.

Bu amaçla bu bölümde sırasıyla

- hata kavramını inceleyerek,
- R ile gösterilen reel sayılar kümesinin bir alt kümesi olan ve $R_f(10)$ ile göstereceğimiz on tabanlı kayan nokta sayı sistemini,
- R den $R_f(10)$ sayı sistemine, yani,

$$R- > R_f(10)$$

dönüşümü ve bu dönüşümde oluşabilen hataları,



- $R_f(2)$ ile göstereceğimiz iki tabanlı kayan nokta sayı sistemini,

- On tabanlı kayan noktalı sayılar ile iki tabanlı kayan nokta sistemi arasındaki,

$$R_f(10) \leftrightarrow R_f(2)$$

  dönüşümü ve bu dönüşümde oluşan hataları,

- İki tabanlı kayan noktalı sayıların 32−bit dönüşümü, yani

$$R_f(2) \rightarrow R_f(2)(32-bit)$$

  veya 64−bit formata dönüşümü, yani,

$$R_f(2) \rightarrow R_f(2)(64-bit)$$

  ve bu dönüşümlerde oluşabilen hataları ve özetle şematik olarak yukarıda ifade edilen ve reel sayılar kümesinden bilgisayar sayı sistemine dönüşüm sürecini ve herbir süreçte oluşabilen hataları inceliyoruz.

- Ayrıca, bilgisayar sayı sisteminden kaynaklanan ve anlam kaybı hatası adı verilen özel bir hatanın nasıl oluştuğunu örnekler üzerinde gözlemliyoruz.

- Yukarıdaki her bir aşamada oluşması mümkün olan hatanın ilerleyen aritmetik işlemlere nasıl yayıldığının incelenmesi de ayrı bir konudur ve bu konuyu Taylor yaklaşımlarıyla ilgili olduğu için bir sonraki bölümün son kısmında inceliyoruz.

Konuyla ilgili detaylar için, faydalandığımız [1],[3], [4],[7],[8] ve [10] nolu kaynakları öneriyoruz.

2.1 Hata(kesme ve yuvarlama hataları)

Ölçüm işlemlerinde genelde gerçek değer yerine, gerçek değeri en yakın biçimde temsil eden yaklaşımı kullanırız. Bu durumda

$$gerçek\ değer = yaklaşım + hata$$

veya

$$hata = gerçek\ değer - yaklaşım$$

bağıntısı ile tanımlanan ve yaklaşımdan kaynaklanan bir hata oluşur ki bu hata sayısal analizde mutlak hata olarak tanımlanır.

TANIM 2.1. *x ile gösterilen bir büyüklüğün gerçek değeri ile bu değeri temsil eden x_f değeri arasındaki fark x_f yaklaşımının* ***mutlak hatası*** *veya kısaca* ***mutlak hata*** *olarak tanımlanır ve*

$$\Delta x = x - x_f$$

notasyonu ile gösterilir.

x_f yaklaşım değeri x gerçek değerinden aşağıda tanımlandığı üzere *kesme* veya *yuvarlama* yapmak suretiyle elde edilebilir:

TANIM 2.2. *$x = d_1.d_2d_3....d_nd_{n+1}...$ değerine, noktadan sonraki n basamakla* ***kesme*** *esasına göre elde edilen yaklaşım $x_f = d_1.d_2d_3....d_n$ olup, bu yaklaşım sonucu oluşan*

$$\Delta x = x - x_f = 0.\overbrace{0..^{n\ adet}..0}\, d_{n+1}\cdots$$

hatasına mutlak kesme(chopping) hatası adı verilir.

Örneğin $x = 1.76$[1] değerine noktadan sonra bir basamakla kesme esasına göre elde edilen yaklaşım $x_f = 1.7$ olup, bu yaklaşım sonucunda oluşan mutlak kesme hata ise

$$\Delta x = x - x_f = 0.06$$

dır.

Güncel bilgisayarlar bu yaklaşım yöntemini kullanmamaktadırlar, dolayısıyla bu bölümde hata kavramı deyince aksi belirtilmedikçe teknik olarak aşağıda tanımlanan ve genelde daha küçük yaklaşım hatasına neden olan *yuvarlama hatasını* ifade ediyor olacağız.

[1]Kesirli sayılarda kullanılan nokta, virgül anlamındadır. Yabancı literatürle uyum açısından virgül yerine nokta kullanıyoruz.

TANIM 2.3. *On tabanlı sistemde* $x = d_0.d_1d_2....d_nd_{n+1}(0 \leq d_i \leq 9)$ *değerine, noktadan sonraki* n *basamakla* ***en yakın sayıya yuvarlama*** *esasına göre elde edilen yaklaşım*

$$x_f = \begin{cases} d_0.d_1d_2....d_n & eğer\ d_{n+1} < 5 \\ d_0.d_1d_2....d_n + (1/10)^n & d_{n+1} \geq 5 \end{cases}$$

olup, bu yaklaşım sonucu oluşan

$$\Delta x = x - x_f$$

hatasına mutlak yuvarlama(round off) hatası adı verilir.

Yukarıdaki örnekte $x = 1.76$ değerine noktadan sonra bir basamakla yuvarlama esasına göre elde edilen yaklaşım

$$x_f = 1.7 + (1/10)^1 = 1.8$$

olup, bu yaklaşım sonucunda oluşan mutlak yuvarlama hatası ise -0.04 tür.

Öteyandan $x = 1.73$ değerine noktadan sonra bir basamakla yuvarlama esasına göre elde edilen yaklaşım

$$x_f = 1.7$$

olup, bu yaklaşım sonucunda oluşan mutlak yuvarlama hatası ise 0.03 tür.

Mutlak hata günlük hayatta algıladığımız hata kavramını tam olarak karşılamaz!

Örneğin $x = 10.1$ birim uzunluğa sahip olan cismin herhangi bir boyutu $x_f = 10$ birim olarak ölçülmüşse, x_f yaklaşımında oluşan mutlak hata veya mutlak yuvarlama hatası

$$\Delta x = x - x_f = 10.1 - 10 = 0.1$$

değerine eşittir.

Öte yandan $x = 1.1$ birim gerçek uzunluğuna sahip olan başka bir cismin boyutu $x_f = 1$ birim olarak ölçülmüşse, x_f nin mutlak hatası

$$\Delta x = 1.1 - 1 = 0.1$$

değerine sahiptir.

Her iki ölçüm sonucunda oluşan mutlak hatalar birbirine eşit olmasına rağmen, her nedense ikinci ölçümde daha fazla hata yaptığımızı düşünürüz. Dolayısıyla mutlak hata kavramının güncel hayatta *algıladığımız hata* kavramını tam olarak karşılamadığını görüyoruz!

Bu durumda güncel algılarımıza uygun bir hata kavramı mevcut olmalıdır ki bu kavram aşağıdaki tanım ile verilmektedir.

TANIM 2.4. *$x \neq 0$ değerine $x_f \neq 0$ ile yuvarlama prensibine göre yaklaşımda oluşan mutlak hatanın x e oranı olarak tanımlanan hataya da x_f nin* ***bağıl yuvarlama hatası*** *veya kısaca* ***bağıl hata*** *adı verilir ve*

$$\varepsilon_b(x) = \frac{\Delta x}{x} \doteq \frac{\Delta x}{x_f}$$

notasyonu ile gösterilir.

Buna göre yukarıdaki birinci ölçüm için oluşan bağıl hata

$$\varepsilon_b(x) = \frac{\Delta x}{x} = \frac{0.1}{10.1} \doteq \frac{0.1}{10} = 0.001$$

ve ikincide oluşan hata ise

$$\varepsilon_b(x) = \frac{\Delta x}{x} = \frac{0.1}{1.1} \doteq \frac{0.1}{1} = 0.1$$

dir, yani ikinci ölçüm sonucu yapılan bağıl hata beklentilerimiz doğrultusunda daha büyüktür. O halde bağıl hata günlük hayatta algıladığımız hata ile daha uyumludur.

Ölçüm sonuçları için yapmak durumunda olduğumuz yaklaşımlara benzer olarak, bilgisayarların sınırlı bellek kapasiteleri ve hız limitleri nedeniyle de, sayısal analiz sürecinde bilgisayarlar üzerinde elimizdeki verilerle çalışmak yerine, onları temsil eden yaklaşımlarla çalışmak durumunda kalırız. Bu durumda kesme veya yuvarlama hataları adını verdiğimiz hatalar oluşur. Sayısal analiz sürecinde oluşabilecek olan kesme veya yuvarlama hatalarını anlayabilmek için öncelikle $R_f(10), R_f(2)$ sayı sistemleri ve $R_f(2)$ sayı sisteminin $R_f(2) - 32$-bit ve $R_f(2) - 64$-bit bellek gösterimlerini yakından incelemeliyiz.

2.2 $R_f(10)$ Kayan nokta sayı sistemi

Bu bölümde $R_f(10)$ sayı sistemini oluşturan sayıların özelliklerini ve $R- > R_f(10)$ dönüşümünde oluşan mutlak ve bağıl kesme ve yuvarlama hatalarını inceliyoruz.

Sonlu sayıda ondalık basamağa sahip herhangi bir x reel sayısı, uygun $m \geq 0, n \geq 0$ tamsayıları için

$$U = \{10^n, 10^{n-1}, \cdots, 10^0, 10^{-1}, \cdots, 10^{-m}\} \tag{2.1}$$

kümesinin elemanlarının lineer bileşimi olarak ifade edilebilir. Diğer bir deyimle, uygun

$$c_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

için

$$\begin{aligned} x &= (c_n c_{n-1} \cdots c_0 . c_{-1} \cdots c_{-m-1} c_{-m})_{10} && (2.2) \\ &= c_n 10^n + c_{n-1} 10^{n-1} + \cdots + c_0 10^0 && (2.3) \\ &\quad + c_{-1} 10^{-1} + \cdots c_{-m-1} 10^{-m-1} + c_{-m} 10^{-m} \end{aligned}$$

biçiminde ifade edilebilir. Burada

$$c_n c_{n-1} \cdots c_0 = c_n 10^n + c_{n-1} 10^{n-1} + \cdots + c_0 10^0$$

sayının tam kısmını,

$$0.c_{-1} \cdots c_{-m-1} c_{-m} = c_{-1} 10^{-1} + \cdots c_{-m-1} 10^{-m-1} + c_{-m} 10^{-m}$$

ise sayının kesir kısmını temsil eder.

x sayısının U kümesinin elemanlarının lineer bileşimi olarak bu şekildeki ifadesine x in onluk sisteme(on tabanlı sisteme) göre gösterimi adı verilmektedir. Örneğin

$$\begin{aligned} &1964.24 \\ &\quad = 1 \times 10^3 + 9 \times 10^2 + 6 \times 10^1 + 4 \times 10^0 + 2 \times 10^{-1} + 4 \times 10^{-2} \end{aligned}$$

olarak $(n = 3, m = -2)$ ifade edilebilir. O halde verilen sayı,

$$U = \{10^3, 10^2, 10^1, 10^0, 10^{-1}, 10^{-2}\}$$

kümesi(veya taban) elemanlarının lineer bileşimi olarak ifade edilmiştir.

En genel olarak $R_f(10)$ sayı sistemi

$$x_f = \pm q_f \times 10^E$$

formatında ifade edilebilen sayılardan oluşur ve bu formatta ifade edilebilen sayılara 10 tabanlı *kayan noktalı sayılar* adı verilir, burada q_f ye, x_f nin **kesir kısmı(veya mantis)** adı verilir ve

$$q_f = 0.d_1d_2...d_n, 0 \leq d_i < 10$$

biçiminde olup, $q_f < 1$ dir. E ise **üs** adı verilen bir tam sayıdır. Böylece 10 tabanlı sistemde bir kayan noktalı sayıyı tanımlayabilmek için *işaret, kesir*(q_f) ve *üs*(E) bilgileri gerekmektedir.

Kesirdeki noktanın pozisyonu, üs değiştirilmek suretiyle sağa veya sola kaydırılabileceğinden ötürü bu formatta ifade edilen sayılara *kayan noktalı sayılar* adı verilmektedir. Örneğin on tabanlı sistemde

$$123.4 = 12.34 \times 10^1 = 1.234 \times 10^2$$

kayan nokta sayıları birbirine eşittir ve bir sayının birden fazla kayan nokta gösterimi mevcuttur . Üs değiştirmek suretiyle kesir noktasının sola doğru nasıl kaydırıldığına dikkat edelim[2].

TANIM 2.5. *$x_f = \pm q_f \times 10^E$ kayan nokta gösteriminde, $q_f = 0.d_1d_2...d_n$ nin d_1 ile gösterilen başlangıç pozisyonundaki rakam sıfırdan farklı ise x_f ye* ***normalize edilmiş sayı*** *adı verilir. Bu durumda $0.1 \leq q_f < 1$ dir. Normalize edilmiş formatta, kayan noktanın sağındaki rakamların sayısına(burada n) x_f nin* ***inceliği****(precision) adı verilir.*

Gözlem 2.1. *Yukarıdaki örnekte görüldüğü üzere aynı sayı birden fazla üs ve mantis ile ifade edilebilir. Diğer değimle en genel halde kayan nokta gösterimi tek türlü değildir. Ancak normalize edilmiş formatta bu gösterim tek türlüdür:*

[2]Kesirli sayıların gösteriminde Türkçe literatürde kullandığımız virgül yerine elektronik ortamda ve yabancı literatüde nokta kullanılır. Biz de uyumluluk açısından nokta kullanıyoruz.

Başlangıç pozisyonundaki rakam, yani kesir noktasının sağındaki ilk rakamın sıfırdan farklı olması durumunda kayan nokta formatı, sayının mantis ve üs değerini tek türlü olarak belirler.

Örneğin

$$123.4 = 0.1234 \times 10^3$$

gösteriminde kesir .1234, taban 10, üs 3 ve sayının inceliği 4 tür. Bu gösterim 123.4 sayısının normalize edilmiş kayan nokta gösterimidir.

2.2.1 $R_f(10)$ sayı sisteminin özellikleri

Kayan nokta sayı sisteminde temsil edilebilecek sayılarının büyüklüklerini ve birbirlerine göre konumlarını yakından görebilmek için bu sayı sistemini yakından incelemek gerekir. Bunun için çok az sayıda nokta içerebilen **Özel** bir kayan nokta sayı sistemi göz önüne alacağız. Özel diyoruz, çünkü hiç bir sistem bu kadar az sayıda nokta içermez. Gerçekçi olmayacak kadar az sayıda nokta içermesine rağmen, örneğimizdeki kayan nokta sistemi ve üzerindeki işlem sonuçları bilgisayarlarda kullanılan gerçek sistemlerin tipik özelliklerini içermektedir.

2.2.2 Özel bir $R_f(10)$ örneği: $\ddot{O}zel\_R_f(10)$

Çok az sayıda eleman içeren ve $\ddot{O}zel\_R_f(10) \subset R$ ile göstereceğimiz on tabanlı bir kayan nokta sayı sistemi gözönüne alalım. Sistemimizin normalize edilmiş

$$x_f = \mp q_f \times 10^E$$

biçiminde sayılardan oluştuğunu ve

$$E = -1, 0, 1$$

ve

$$q_f = 0.d_1d_2, d_1 \neq 0, d_2 = 0, 1, \ldots, 9$$

değerlerini alabileceğini kabul edelim. Teknik bir ifade ile

$$\ddot{O}zel\_\ R_f(10) = \left\{ \begin{array}{c} x_f = \mp q_f \times 10^E |\ q_f = 0.d_1d_2, d_i = 0, 1, \ldots, 9; \\ i = 0, 1; d_1 \neq 0; E = 0, 1, 2 \end{array} \right\}$$

noktalarında oluşmaktadır.

*Özel*_ $R_f(10)$ kayan nokta kümesinin elemanlarını belirleyerek, sayıların birbirlerine göre konumlarını ve *Özel*_ $R_f(10)$ üzerindeki aritmetik işlemlerin özelliklerini incelemeye çalışalım:

- *Özel*_ $R_f(10)$ da temsil edilebilecek en büyük sayı $d_1 = 9, d_2 = 9$ ve $E = 1$ ile pozitif işaretli $x_f = 0.99 \times 10^2 = 99$ sayısıdır. O halde negatif işareti de dikkate aldığımızda

$$\ddot{O}zel\_R_f(10) \ \subset [-99, 99]$$

olduğunu görürüz.

- Örnek sistemde her bir üs için temsil edilebilecek pozitif sayılar aşağıdaki tablolarda gösterilmektedir.

$E = 0$					
	$d_1 = 1 \rightarrow$	0.10	0.11	$\cdots$	0.19
	$d_1 = 2 \rightarrow$	0.20	0.21	$\cdots$	0.29
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	$d_1 = 9 \rightarrow$	0.90	0.91	$\cdots$	0.99

$E = 1$					
	$d_1 = 1 \rightarrow$	1.0	1.1	$\cdots$	1.9
	$d_1 = 2 \rightarrow$	2.0	2.1	$\cdots$	2.9
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	$d_1 = 9 \rightarrow$	9.0	9.1	$\cdots$	9.9

$E = 2$					
	$d_1 = 1 \rightarrow$	10	11	$\cdots$	19
	$d_1 = 2 \rightarrow$	20	21	$\cdots$	29
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	$d_1 = 9 \rightarrow$	90	91	$\cdots$	99

Her bir tabloda 90 adet olmak üzere *Özel*_ $R_f(10)$ da 270 adet pozitif sayının yer aldığını görürüz. Negatif sayıların da ilavesiyle örnek kayan nokta sayı sistemimiz toplam 540 adet normalize edilmiş sayıdan oluşmaktadır.

- *Özel_* $R_f(10)$ da komşu noktalar arasındaki uzaklığın eşit olmadığını ve orjinden uzaklaştıkça komşu noktalar arasındaki mesafenin E nin her bir yeni değeri için arttığını gözlemliyoruz. Ancak E nin aynı değeri için elde edilen ve birbirine komşu olan kayan nokta sayıları arasındaki uzaklığın eşit olduğunu görüyoruz. Örneğin $E = 0$ için komşu sayılar arasındaki uzaklık 0.01, $E = 1$ için komşu sayılar arasındaki uzaklık 0.1 ve $E = 2$ için komşu sayılar arasındaki uzaklık 1 dir ve E nin her bir değeri için elde edilen komşu sayılar arasındaki uzaklık, bir öncekilerin 10(yani taban) katıdır.

- Sıfır sayısının normalize edilmiş bir kayan nokta sayısı olmadığını görüyoruz. $d_1 = 0$ değerine izin verilmesi durumunda sıfır sayısını da sistemde temsil edebiliriz. $d_1 = 0$ değerine karşılık gelen normalize edilmiş formatta ifade edilemeyen sayılara **subnormal** sayılar adı verilmektedir. Bu durumda sistemimizin subnormal sayıları $0.01, 0.02, ..., 0.09$ ve bu sayıların negatifleri ile birlikte $0.0, -0.0$ sayısı olmak üzere toplam 20 adet sayıdır. Burada 0.0 ve -0.0 ın her ikisi de sıfır sayısını temsil etmektedir.

- Temsil edilebilecek en büyük pozitif sayıdan daha büyük bir sayı için **inf**(∞) ve en küçük negatif sayıdan daha küçük bir sayı durumunda $-$**inf**$(-\infty)$ sembolleri uygun subnormal formatta ifade edilirler. Ayrıca reel sayılarda belirsizlik durumları olarak bilinen $0/0$, ∞/∞ için **NaN**(**N**ot **a N**umber, sayı değil) gösterimi kullanılır. Örnek sayı sistemimizde

$$99 + 1 = \text{inf}, -99 - 1 = -\text{inf}, 0/0 = \text{NaN}$$

  olarak sembolize edilir. Bu semboller için bellekte özel gösterimler kullanılır.

- *Özel_* $R_f(10)$ *da kesme ve yuvarlama:* *Özel_* $R_f(10)$ sayı sisteminde herhangi iki sayı ile gerçekleştirilen aritmetik işlem sonucu *Özel_* $R_f(10)$ sayı sisteminin bir elemanı değilse(çok büyük ve küçük sayı olma durumu hariç), bu durumda elde edilen sonuç en yakın *Özel_* $R_f(10)$ sayısına **kesme** veya **yuvarlama** prensibine göre yuvarlanır. *Kesme prensibine* göre yuvarlama işlemi aşağıda görüldüğü biçimde noktadan sonraki kısmın kesilmesi suretiyle gerçekleştirilir:

$$10.4 \doteq 10, 10.5 \doteq 10, 10.7 \doteq 10$$

Yuvarlama prensibinde ise virgülden(veya noktadan) sonraki kısım doğrudan kesilmeyerek, virgülden(veya noktadan) sonraki ilk rakamın tabanın yarısından büyük veya eşit olması veya küçük olması durumlarında farklı yuvarlama işlemleri gerçekleştirilir. Aşağıdaki örnekleri inceleyelim:

$$10.4 \doteq 10, 10.5 \doteq 11, 10.7 \doteq 11$$

Şimdi de yuvarlama işlemlerinin $\ddot{O}zel\_R_f(10)$ üzerindeki aritmetiği nasıl etkilediğini inceleyelim:

- $\ddot{O}zel\_R_f(10)$ *da toplama işlemine göre birleşme özelliği geçerli değildir*: *İşlemlerin en yakın sayıya yuvarlama prensibine göre* gerçekleştirildiğini kabul edersek

  $$(0.3 + 0.4) + 10 = 0.7 + 10 \doteq 11 \neq 10 \doteq 0.3 + (0.4 + 10)$$

  elde ederiz. Çünkü 10.7 sayı sistemimizin bir elemanı değildir ve sistemde bu sayıya en yakın sayı 11 dir. Benzer biçimde $10.4 \doteq 10$ dur.

  Eğer *işlemlerin kesme prensibine göre* gerçekleştirildiğini kabul edersek

  $$(0.6 + 0.7) + 10 = 11 \neq 10 = 0.6 + (0.7 + 10)$$

  elde ederiz, çünkü kesme prensibine göre $11.3 \doteq 11$ ve $10.7 \doteq 10$, $10.6 \doteq 10$ dur.

- $\ddot{O}zel\_R_f(10)$ *toplama işlemine göre de kapalı değildir*: Örneğimiz için

  $$50, 60 \in \ddot{O}zel\_R_f(10)$$

  fakat

  $$110 \notin \ddot{O}zel\_R_f(10)$$

  dur.

- $R -> \ddot{O}zel\_R_f(10)$ *dönüşümü ve hata*: $x \in R \backslash R_f(10) \subset [-99, 99]$ olması durumunda, x yerine $x_f \in R_f(10)$ yaklaşımı kullanılır. Söz konusu yaklaşım kesme veya en yakın sayıya yuvarlama esasına göre gerçekleştirilir: $x = 11.6$ sayısı gözönüne aldığımız örnek kayan nokta

sayı sisteminde temsil edilmemektedir. Bu durumda kesme prensibine göre $x_f = 11$ yaklaşımı kullanılır. Bu durumda oluşan mutlak hata

$$\Delta x = 11.6 - 11 = 0.6$$

ve bağıl hata ise

$$\epsilon_b(x) = \frac{\Delta x}{x} = \frac{0.6}{11.6} \doteq 0.0517$$

dir. En yakın sayıya yuvarlama prensibine göre ise kesir kısım atılarak, bir öndeki rakama 1 ilave etmek suretiyle $x_f = 12$ alınır. Bu durumda oluşan mutlak hata

$$\Delta x = 11.6 - 12 = -0.4$$

ve bağıl hata ise

$$\epsilon_b(x) = \frac{\Delta x}{x} = \frac{-0.4}{11.6} \doteq -0.0345$$

dir.

Şimdi de gerçek bir kayan nokta sistemine dönüşümde oluşabilen hataları inceleyelim:

2.2.3 $R- > R_f(10)$ dönüşümü ve hata

$x \in R$ yerine $x_f \in R_f$ yaklaşımının alınması durumunda yuvarlama hataları oluşur:

- – 10 tabanlı sistemde

  $$x = (0.d_1 \cdots d_n d_{n+1} \cdots) \times 10^E, d_1 \neq 0, 0 \leq d_i \leq 9, i = 1, 2, ..., 9$$

  sayısı için *kesme yöntemine göre* R_f deki yaklaşım

  $$x_f = (0.d_1 \cdots d_n) \times 10^E$$

  olarak tanımlanır. Bu durumda oluşan mutlak hatanın üst sınırı

  $$\begin{aligned} \Delta x &= x - x_f \\ &= (0.0...0 d_{n+1} \cdots) \times 10^E \\ &= (d_{n+1}.d_{n+2} \cdots) \times 10^{-(n+1)} \times 10^E \\ &\leq 10 \times 10^{-n-1} \times 10^E = 10^{E-n} \end{aligned}$$

olarak tahmin edilebilir(burada $d_{n+1}.d_{n+2}\cdots \leq 10$ eşitsizliğini kullandık).

– Yine 10 tabanlı sistemde

$$x = (0.d_1 \cdots d_n d_{n+1} \cdots) \times 10^E, 0 \leq d_i \leq 9, d_1 \neq 0, i = 1, 2, ..., 9$$

sayısı için *en yakın sayıya yuvarlama esasına göre* R_f de

$$x_f = \begin{cases} (0.d_1 \cdots d_n) \times 10^E, & d_{n+1} \leq 4 \\ (0.d_1 \cdots d_n + 10^{-n}) \times 10^E & d_{n+1} \geq 5 \end{cases}$$

olarak tanımlanır. Örneğin $d_{n+1} \leq 4$ olması durumunda

$$\begin{aligned} \Delta x &= x - x_f \\ &= (0.0_1 \cdots 0_n d_{n+1} \cdots) \times 10^E \\ &\leq (d_{n+1}.d_{n+2} \cdots) \times 10^{-(n+1)+E} \\ &\leq 5 \times 10^{-(n+1)+E} = \frac{1}{2} \times 10^{E-n} \end{aligned}$$

olarak tahmin edilebilir.$d_{n+1} \geq 5$ olması durumunda ise

$$\begin{aligned} \Delta x &= x - x_f \\ &= (0.d_1 \cdots d_n d_{n+1} \cdots) \times 10^E - (0.d_1 \cdots d_n + 10^{-n}) \times 10^E \\ &= (0.0_1 \cdots 0_n d_{n+1} \cdots) \times 10^E - 1 \times 10^{E-n} \\ &= (0.d_{n+1} \cdots) \times 10^{E-n} - 10^{E-n} \\ &= -[1 - (0.d_{n+1} \cdots)] \times 10^{E-n} \end{aligned}$$

olup,

$$|\Delta x| = [1 - (0.d_{n+1} \cdots)] \times 10^{E-n} \leq \frac{1}{2} \times 10^{E-n} \tag{2.4}$$

elde ederiz. O halde en yakın sayıya göre yuvarlama esasına göre d_{n+1} değerinden bağımsız olarak

$$|\Delta x| \leq \frac{1}{2} \times 10^{E-n}$$

elde ederiz.

– Öte yandan yukarıda incelenen yuvarlama işlemin gerçekleştirilme biçiminden bağımsız olarak

$$\begin{aligned} |\varepsilon_b(x)| &= \frac{|\Delta x|}{|x|} \\ &\leq \frac{\frac{1}{2}\times 10^{E-n}}{(0.d_1\cdots d_n d_{n+1}\cdots)\times 10^E} \\ &\leq \frac{1}{2}\times 10^{-n+1} = 5\times 10^{-n} \end{aligned} \tag{2.5}$$

elde ederiz çünkü $d_1 \geq 1$ olduğundan

$$0.d_1\cdots d_n d_{n+1}\cdots >= 10^{-1}$$

dir. Bağıl hata için elde ettiğimiz üst sınırı $\epsilon_{\max}$ ile göstereceğiz. O halde 10 tabanlı sayı sistemi için

$$\epsilon_{\max} = 5\times 10^{-n}$$

dir.

– 2 tabalı sistem için

$$\epsilon_{\max} = \frac{1}{2}\times 2^{-n+1} = 2^{-n}$$

olarak elde edililir ve bu değer *bilgisayar* epsilonu olarak adlandırılır: $|\varepsilon_b(x)| \leq \epsilon_{\max}$ dır.

ÖRNEK 2.1.

$$\begin{aligned} x &= 3.14159265358979 \\ &= 0.314159265358979\times 10^1 \end{aligned}$$

için en yakın sayıya yuvarlama prensibine göre elde edilen

$$x_f \doteq 0.31416\times 10^1$$

yaklaşımı ile oluşan mutlak ve bağıl hatayı belirleyiniz ve (2.4) ,(2.5) ile verilen üst sınırların geçerli olduğunu gözlemleyiniz.

Çözüm.

$$\begin{aligned}\Delta x &= x - x_f \\ &= 7.34641020683213 \times 10^{-6} \\ &= 0.0734641020683213 \times 10^{-4} \\ &\leq \frac{1}{2} \times 10^{1-5} = \frac{1}{2} \times 10^{E-n}\end{aligned}$$

ve

$$\begin{aligned}\varepsilon_b(x) &= \frac{\Delta x}{x} \\ &\leq \frac{\frac{1}{2} \times 10^{1-5}}{0.314159265358979 \times 10^{1}} \\ &< \frac{1}{2} \times 10^{-4}\end{aligned}$$

olup, (2.4) ,(2.5) ile verilen üst sınırlar geçerlidir.

Bir diğer hata kaynağı on tabanlı sistemden iki tabanlı sisteme dönüşümde oluşan hatalardır.

2.3 $R_f(10) \longleftrightarrow R_f(2)$ Taban dönüşümleri ve ilgili hatalar

Bilgisayarların çoğu iki tabanlı sayı sistemini kullanırlar. Kullandığımız on tabanlı sayı sisteminde sonlu sayıda rakamla temsil edilebilen bir sayının bilgisayarların kullandıkları iki tabanlı sayı sisteminde sonlu sayıda rakamla ifade edilememe ihtimali söz konusudur.

Bu noktayı açıklamak için öncelikle göz önüne aldığımız bilgisayar sisteminin iki tabanlı sayı sistemini kullandığını kabul ederek, iki tabanlı ve on tabanlı sayı sistemleri arasındaki dönüşümün nasıl gerçekleştirildiğini inceleyelim:

2.3.1 $R_f(2)$ ve $R_f(2) -> R_f(10)$ dönüşümü

İkili sistemde *sonlu sayıda* basamağa sahip herhangi bir

$$x = (c_n c_{n-1} \cdots c_0 . c_{-1} c_{-2} \cdots c_{-m})_2$$

reel sayısı, uygun $m \geq 0, n \geq 0$ tamsayıları için (2.1) e benzer olarak

$$V = \{2^n, 2^{n-1}, \cdots, 2^0, 2^{-1}, \cdots, 2^{-m}\} \tag{2.6}$$

kümesinin, veya taban elemanlarını, elemanlarının lineer kombinasyonu olarak ifade edilebilir.

Diğer değimle uygun $c_i \in \{0, 1\}$ ve $n \geq 0, m \geq 0$ tamsayıları için

$$\begin{aligned} x &= (c_n c_{n-1} \cdots c_0 . c_{-1} c_{-2} \cdots c_{-m})_2 \\ &= c_n 2^n + c_{n-1} 2^{n-1} + \cdots + c_0 2^0 \\ &\quad + c_{-1} 2^{-1} + \cdots + c_{-m-1} 2^{-m-1} + c_{-m} 2^{-m} \end{aligned}$$

biçiminde V kümesinin elemanlarının lineer bileşimi olarak yazılabilir.

$R_f(2)$ ile gösterdiğimiz iki tabanlı sistemde kullanılan rakamlar 0 ve 1 olup, taban ise 2 dir.

İkili sayı sisteminde herhangi bir sayı $c_i \in \{0, 1\}$ rakamları ve 2 nin azalan kuvvetleri cinsinden ifade edilerek on tabanlı sisteme dönüştürülür. Örneğin

$$\begin{aligned} (101101)_2 &= 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \\ &= 32 + 8 + 4 + 1 \\ &= 4 \times 10^1 + 5 \times 10^0 \\ &= (45)_{10} \end{aligned}$$

İki tabanlı sistemdeki kesirli bir sayı da benzer biçimde on tabanlı sisteme dönüştürülebilir. Örneğin

$$\begin{aligned} (0.101111)_2 \times 2^3 &= (101.111)_2 \\ &= 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} \\ &= 4 + 1 + 1/2 + 1/4 + 1/8 = 5.875. \end{aligned}$$

olarak elde edilir.

2.3.2 $R_f(10) -> R_f(2)$ dönüşümü

- *Tamsayı dönüşümü:* Bu dönüşüm için aşağıdaki örneği inceleyelim.

ÖRNEK 2.2. 1964 *sayısını iki tabanlı gösterimini elde ediniz.*

Çözüm.

$$2^{10} = 1024$$

olduğu dikkate alındığında

$$1964 = c_{10}2^{10} + c_9 2^9 + \cdots c_1 2^1 + c_0 2^0$$

sağlanacak biçimdeki

$$c_i \in \{0, 1\}, i = 0, 1, ..., 10$$

sabitlerini belirlemeliyiz. En son sabit olan c_0 ın dönüştürülmek istenen sayının ikiye bölümünde elde edilen kalan olduğu açıktır. Yani

$$1964 = 2 \times 982 + c_0$$

dır ve 1964 çift sayı olduğu için $c_0 = 0$ dır. Şimdi ise eşitliğin her iki tarafını ikiye bölerek elde edilen

$$982 = c_{10}2^9 + c_9 2^8 + \cdots + c_2 2^1 + c_1$$

ifadesinden de c_1 in 982 nin 2 ye bölümünden elde edilen kalan olduğuna dikkat edelim. Yani

$$982 = 2 \times 491 + c_1$$

dir. Benzer işlemler tekrar edilerek

$$\begin{aligned}
1964 &= 982 \times 2 + c_0 \Rightarrow c_0 = 0 \\
982 &= 491 \times 2 + c_1 \Rightarrow c_1 = 0 \\
491 &= 245 \times 2 + c_2 \Rightarrow c_2 = 1 \\
245 &= 122 \times 2 + c_3 \Rightarrow c_3 = 1 \\
122 &= 61 \times 2 + c_4 \Rightarrow c_4 = 0 \\
61 &= 30 \times 2 + c_5 \Rightarrow c_5 = 1 \\
30 &= 15 \times 2 + c_6 \Rightarrow c_6 = 0 \\
15 &= 7 \times 2 + c_7 \Rightarrow c_7 = 1 \\
7 &= 3 \times 2 + c_8 \Rightarrow c_8 = 1 \\
3 &= 1 \times 2 + c_9 \Rightarrow c_9 = 1 \\
1 &= 0 \times 2 + c_{10} \Rightarrow c_{10} = 1
\end{aligned}$$

elde edilir. O halde

$$\begin{aligned} 1964 &= 1\times 10^3 + 9\times 10^2 + 6\times 10^1 + 4\times 10^0 \\ &= c_{10}2^{10} + c_9 2^9 + \cdots + c_2 2^2 + c_1 2^1 + c_0 \\ &= 1\times 2^{10} + 1\times 2^9 + 1\times 2^8 + 1\times 2^7 \\ &+ 0\times 2^6 + 1\times 2^5 + 0\times 2^4 + 1\times 2^3 + 1\times 2^2 \\ &+ 1\times 2^3 + 1\times 2^2 + 0\times 2^1 + 0\times 2^0 \end{aligned}$$

olup,

$$1964 = (11110101100)_2$$

iki tabanlı gösterimi elde edilir.

Özetle on tabanlı sayıyı iki tabanlı bir sayıya dönüştürmek için uygulamamız gereken pratik kural şudur:

Sayıyı ikiye bölerek kalanı not ettikten sonra bölüm kısmı ile aynı işlemi tekrarlanır. Elde edilen kalanların sondan başa doğru yan yana dizilimi ilgili sayının iki tabanlı gösterimidir.

Program 2.1 yukarıda açıklanan yöntemi kullanmak suretiyle, girilen on tabanlı pozitif bir sayının iki tabanlı gösterimini belirler:

Yukarıda belirtilen yöntem ve verilen programı MATLAB/OCTAVE ortamında çalıştırarak aşağıdaki tabloda verilen dönüşümleri elde ederiz:

On tabanlı sayı	İki tabanlı gösterimi
2	10
3	11
4	100
5	101
10	1010
100	1100100
1000	1111101000

- *Kesirli sayı dönüşümü:* On tabanlı sistemde ifade edilen herhangi bir kesirli sayı 10 nun negatif kuvvetleri yardımıyla ifade edilebileceği gibi, iki tabanlı sayı sisteminde de 2 nin negatif kuvvetleri yardımıyla ifade edilebilir: Kesirli sayı dönüşümünde $[[x]]$ notasyonu ile x sayısının tam kısmını ve $\{x\}$ ile de kesirli kısmını gösterelim. Kesirli sayı taban dönüşümünü aşağıdaki örnek üzerinde inceleyelim:

```
%-------------------------------------------------------
function gikili=ondaniki(sayi)
% On tabanlı sistemden iki tabanlı sisteme dönüştürür

 sayac=1;
  while sayi>=2
    kalan=mod(sayi,2);
    ikili(sayac)=kalan; % kalanlar
    sayi=(sayi-kalan)/2;
    sayac=sayac+1;
  end
  ikili(sayac)=sayi;
  m=length(ikili);
  for i=1:m
    gikili(i)=ikili(m+1-i); %kalanlarin tersten dizilimi
  end
%-------------------------------------------------------
```

Program 2.1: On tabanlı sistemden iki tabanlı sisteme dönüşüm

ÖRNEK 2.3. *$A = 0.125$ sayısının ikili sistemdeki gösterimini belirleyiniz.*

Çözüm.

On tabanlı sistemde

$$A = 0.125 = 1 \times 10^{-1} + 2 \times 10^{-2} + 5 \times 10^{-3}$$

olarak ifade edilir. Aynı sayıyı iki tabanlı sistemde

$$\begin{aligned} A &= 0.125 = (0.d_1d_2d_3d_4\ldots) \\ &= d_1 \times 2^{-1} + d_2 \times 2^{-2} + d_3 \times 2^{-3} + d_4 \times 2^{-4} + \cdots \end{aligned}$$

şeklinde ifade ettiğimizi varsayalım. İfadenin her iki yanını 2 ile çarparsak

$$2A = 0.250 = d_1 + d_2 \times 2^{-1} + d_3 \times 2^{-2} + d_4 \times 2^{-3} + \cdots$$

elde ederiz. Burada

$$d_1 = [[2A]] = 0$$

ve

$$k_1 = \{2A\} = 0.250 = d_2 \times 2^{-1} + d_3 \times 2^{-2} + d_4 \times 2^{-3} + \cdots$$

dir. Her iki yanı tekrar 2 ile çarparak

$$2k_1 = 0.50 = d_2 + d_3 \times 2^{-1} + d_4 \times 2^{-2} + \cdots$$

$$d_2 = [[2k_1]] = 0$$

dır. Benzer biçimde

$$k_2 = \{2k_1\} = 0.50 = d_3 \times 2^{-1} + d_4 \times 2^{-2} + \cdots$$

$$2k_2 = 1.0 = d_3 + d_4 \times 2^{-1} + \cdots$$

$$d_3 = [[2k_2]] = 1$$

ve

$$k_3 = \{2k_2\} = 0$$

elde edilir. Böylece kesir kısmı sıfır olana kadar işleme devam ettirilerek elde edilen $d_i, i = 1, 2, \ldots$değerleri kaydedilir. O halde

$$A = 0.125 = (0.001)_2$$

iki tabanlı gösterimi elde edilir.

ÖRNEK 2.4. 1964.125 *sayısının iki tabanlı gösterimini elde ediniz.*

Çözüm.

Bunun için yapmamız gereken işlem, yukarıdaki iki örneğe ait sonuçları birleştirmektir. On tabanlı sistemde

$$\begin{aligned} 1964.125 &= 1 \times 10^3 + 9 \times 10^2 + 6 \times 10^1 + 4 \times 10^0 \\ &\quad +1 \times 10^{-1} + 2 \times 10^{-2} + 5 \times 10^{-3} \end{aligned}$$

olacak şekilde ifade edildiği gibi, tam ve kesirli kısımlara karşılık gelen iki tabanlı gösterimler de birleştirildiğinde

$$\begin{aligned} 1964.125 &= (1111010100.001)_2 \\ &= (1.111010100001)_2 \times 2^9 \end{aligned}$$

normalize edilmiş kayan nokta gösterimi elde edilir.

Böylece on tabanlı sistemdeki kesirli bir sayının ikili sistemdeki karşılığını nasıl elde edeceğimizi öğrenmiş bulunuyoruz. Şimdi de $R_f(2)$ sayı sisteminin elemanlarının bilgisayar belleğinde nasıl temsil edildiklerini inceleyelim.

2.4 $R_f(2)$ nin bellek gösterimi

$IEEE754$ standartı adı verilen uluslararası standarta göre x_f ile gösterilen bir kayan nokta sayısı, 32 bit formatta

1 bit(üs işareti)	8 bit(Üs)	23 bit (Mantis)

ve 64 bit formatta ise

1 bit(üs işareti)	11 bit(Üs)	52 bit (Mantis)

bellek alanlarında temsil edilir. MATLAB/Octave 64 bit formatı kullanır[8].

İşaret bitindeki '0', ilgili sayının pozitif ve '1' ise negatif olması anlamına gelir.

İkili sistemde 32 bit formatında temsil edilebilecek normalize edilmiş

$$x_f = \pm(1.d_1d_2\ldots d_{23}) \times 2^e$$

sayısını göz önüne alalım. Kapalı bit gösterimi(implicit bit representation) adı verilen bir standarta göre başlangıç rakamı olan 1 sisteme kaydedilmez.

Orjine göre simetrik olmayan(biased) formatta negatif üsleri temsil etmek için işaret biti kullanılmaz ve

Gerçek üs=depolanan üs-*127*

kuralına göre gerçek üs değerleri elde edilir. Burada 127 ise öteleme sabiti olarak adlandırılır. Bu kurala göre aşağıdaki tabloda belirtilen depolanan üsler, karşılarındaki gerçek üs değerlerini temsil ederler:

depolanan üs	gerçek üs	depolanan üs	gerçek üs
$1 \rightarrow$	-126	$128 \rightarrow$	1
$2 \rightarrow$	-125	$129 \rightarrow$	2
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$127 \rightarrow$	0	$254 \rightarrow$	127

ÖRNEK 2.5.

$$1.964125 \times 10^3 = (1.111010100001)_2 \times 2^9$$

sayısının 32 bit formatta bellek alanına yerleşimini belirleyiniz.

Çözüm.

Yukarıdaki tabloya göre verilen sayının üssü olan 9 değeri 136 depolanan üs değerine karşılık gelmektedir. Öte yandan

$$136 = (10001000)_2$$

dir. Mantisin başlangıç bitindeki 1 değeri kapalı bit gösterimine göre depolanmaz. Kesir noktası da bellek gösteriminde yer almaz. Sadece kesir kısmı olan

$$11101010000$$

1 değeri 23 bitlik alana sağa yaslı olarak depolanır. Sayı pozitif olduğu için işaret biti 0 olmalıdır. Dolayısıyla

0	*1*	*0*	*0*	*0*	*1*	*0*	*0*	*0*	**0**	1	1	1	0	1	0	1	0	0	0	0	1

gösterimini elde ederiz, burada **0**= 0000000000 dır.

2.4.1 $R_f(2) - 32$-bit sisteminde en büyük ve en küçük sayılar ve MATLAB/Octave gösterimleri

Bu formatta 8 bitlik üs alanında ikili sisteme göre temsil edilebilecek en büyük üs

$$\begin{aligned} e &= (11111110)_2 \\ &= 0 \times 2^0 + 1 \times 2^1 + 1 \times 2^2 + 1 \times 2^3 \\ &\quad +1 \times 2^4 + 1 \times 2^5 + 1 \times 2^6 + 1 \times 2^7 \\ &= 254 \end{aligned}$$

tür ve yukarıdaki tabloya göre depolanan bu değer 127 gerçek üs değerini temsil eder.

Ayrıca en büyük mantis değeri

$$\begin{aligned} (1.1_1 1_2 \ldots 1_{23})_2 &= 1 + 1/2^1 + 1/2^2 + \cdots + 1/2^{23} \\ &= \frac{1 - (1/2)^{24}}{1 - 1/2} = (2 - 2^{-23}) \end{aligned}$$

olup, en büyük pozitif kayan noktalı sayı

$$\begin{aligned} x_f &= (2-2^{-23})\times 2^{127} = 3.402823466385289e+038 \\ &= \boxed{0}\boxed{11111110}\boxed{111...1} \end{aligned}$$

dir.

Bu sayı MATLAB/Octave ortamında

>> realmax('single')

ans =

3.4028235e + 038

olarak elde edilir.

Benzer olarak en küçük pozitif değer ise üssün mutlak değerce alabileceği en büyük negatif sayıya(-126) karşılık gelir.-126 için depolanan değer, yukarıdaki tablodan 1 e eşittir. O halde

$$e = (00000001)_2$$

olup,

$$x_f = 2^{-126} = 1.175494350822288e-038$$

sayısının 32-bit formatta bellek gösterimi

0	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	**0**

olarak elde edilir, burada $\mathbf{0} = (0_1 0_2 ... 0_{23})$ dür.

Bu sayı MATLAB/Octave ortamında

>> realmin('single')

ans =

1.1754944e − 038

olarak elde edilir.

Benzer biçimde

0	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	**0**

bellek gösterimi ∞, ve

1	*1*	*1*	*1*	*1*	*1*	*1*	*1*	*1*	**0**

ise $-\infty$ sembolünü temsil eder. $e = (11111111)_2$ ve mantis kısmında herhangi bir bit değerinin sıfırdan farklı olacak biçimde depolanan değer ise **NaN** sembolünü temsil eder.

2.4.2 $R_f(2) - 64$-bit sisteminde en büyük ve en küçük sayılar ve MATLAB/Octave gösterimleri

32−bit formatında olduğu gibi orjine göre simetrik olmayan(biased) formatta negatif üsleri temsil etmek için işaret biti kullanılmaz, ancak bu durumda

$$\text{Gerçek üs=depolanan üs} - 1023$$

kuralına göre gerçek üs değerleri elde edilir. Burada 1023 ise öteleme sabiti olarak adlandırılır. Bu kurala göre aşağıdaki tabloda belirtilen depolanan üsler, karşılarındaki gerçek üs değerlerini temsil ederler:

depolanan üs	gerçek üs	depolanan üs	gerçek üs
$1 \rightarrow$	-1022	$1024 \rightarrow$	1
$2 \rightarrow$	-1021	$1025 \rightarrow$	2
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$1023 \rightarrow$	0	$2046 \rightarrow$	1023

Pozitif bir sayı için 64−bit formatta 11 bitlik üs alanında ikili sisteme göre temsil edilebilecek en büyük üs

$$e = (11111111110)_2 = (2046)_{10} \rightarrow 1023$$

ve en küçük üs ise

$$e = (00000000001)_2 = (1)_{10} \rightarrow -1022$$

dir. Ayrıca en büyük mantis değeri, 2 den bir önceki sayı olan

$$\begin{aligned}(1.1_1 1_2 \cdots 1_{52})_2 &= 1 + 1/2^1 + 1/2^2 + \cdots + 1/2^{52} \\ &= \frac{1-(1/2)^{53}}{1-1/2} = (2 - 2^{-52})\end{aligned}$$

değeridir ve en büyük pozitif kayan noktalı sayı ise

$$x_f = (2 - 2^{-52}) \times 2^{1023} = 1.797693134862316e + 308$$

olup bu sayının bellek gösterimi

0	$0_1 0_2 ... 0_{10} 1_{11}$	$1_1 1_2 1_3 ... 1_{52}$

elde edilir.

Bu sayı MATLAB/Octave ortamında yukarıda belirtildiği üzere

>> realmax
ans =
1.797693134862316e + 308
olarak elde edilir.

Benzer olarak en küçük pozitif değer ise üssün alabileceği mutlak değerce en büyük negatif sayıya(-1022) karşılık gelir ve

$$x_f = 2^{-1022} = 2.225073858507201e - 308$$

olup, ikili sistem karşılığı

0	0000000001	**0**

dır, burada $\mathbf{0} = (0_1 0_2 ... 0_{52})$ dir. Bu sayı yukarıda da belirtildiği gibi MATLAB/Octave ortamında

>> realmin
ans =
2.225073858507201e − 308
olarak elde edilir. $\mp\infty$ ve NaN gösterimleri $32-$bit formatta olduğu gibidir.

2.4.3 MATLAB/Octave üzerinde $R_f(2) - 64$ bit aritmetiği ve oluşabilen hatalar

Bilgisayarların kullandığı kayan nokta sayı sistemi de yukarıda örnekte özelliklerini incelediğimiz Özel sistemin özelliklerini taşır. Bu özellikleri bu dökümanı hazırladığımız bilgisayarın kullandığı kayan nokta sayı sistemi üzerinde ve MATLAB/Octave üzerinde kısaca inceleyelim:

- $R_f(2) - 64$ *-bit sonlu sayıda elemandan oluşur:*

  $R_f(2) - 64$ sisteminde, kullandığımız bilgisayar

  $$[-1.7977E + 308, 1.7977E + 308]$$

  aralığı içerisindeki kayan nokta sayılarından oluşan sonlu bir sayı kümesi ile çalışır. Herhangi bir işlem sonucunun bu aralıkta yer alan sayıdan daha büyük bir sayı olması durumunda sonuç *sonsuz=inf* olarak yorumlanır. Örneğin MATLAB/Octave ortamında

$>>1.797693134862316E+308+1E292$

$$ans = \inf$$

sonsuz(inf-infinity) olarak yorumlanmaktadır. Bilgisayarımızda yer alan ve sıfırdan büyük normalize edilmiş ilk pozitif sayıyı ise *realmin* komutuyla belirlenebilir:

- >> x=realmin

  $x = 2.2251E-308$.

- Sıfır ile kayan nokta sayı sisteminde yer alan bu en küçük sayı arasında oluşan bir işlem sonucu bazı programlarda çok küçük sayı (*underflow*) hatası olarak algılanır ve genelde sonuç sıfır olarak kabul edilerek işleme devam edilir. Bazı uygulamalar genişletilmiş kayan nokta sistemi adı verilen genişletilmiş sistemi kullanarak $[0, realmin]$ aralığında sonlu sayıda *denormal* veya *subnormal* adı verilen sayının temsiline de izin veriler[7]. Bu işlemi $q_f = d_0.d_1d_2...d_n$ gösteriminde d_0 ın da sıfıra eşit olmasına izin vermek suretiyle gerçekleştirirler. Örneğin MATLAB/Octave ortamında *realmin* ile temsil edilebilen sayıdan daha küçük subnormal sayılara izin verilmektedir:

$>> 1e-315$

ans =

1.0000e − 315

Fakat

$>> 1e-324$

ans =

0

dır. Bunlara ilaveten belirsizlik adını verdiğimiz tanımlı olmayan (0/0, ∞/∞ gibi) bir işlem sonucu ise NaN(Not a Number- sayı değil) olarak kabul edilir.

$>> 0/0$

ans =

NaN

- *Hiç bir irrasyonel sayı veya devirli rasyonel sayı R_f (10) da yer almaz(Her bir eleman için sınırlı bellek alanı):*

  $\pi, e, \sqrt{2}$ gibi hiç bir reel sayı sonlu basamaklı q_f ile $x_f = \pm q_f \times 10^E$ biçiminde ifade edilemez. Bu durumda irrasyonel sayılarla işlem yapmak yerine, bu sayılara en yakın olan ve R_f (10) da temsil edilebilen sayılarla çalışılır ve sonuç olarak yuvarlama hatası oluşur. Öte yandan $1/3 = 0.\overline{3}$ devirli sayısı da R_f (10) da yer almaz çünkü sonlu basamaklı bir mantise sahip değildir.

  MATLAB/Octave ortamında

  $>>$ sin(pi)

  ans =

  $1.2246e-016$ elde edilir, sıfıra eşit olması beklenen bu sonucun sebebi R_f (10) daki pi nin tam olarak R deki π ye eşit olmamasıdır.

- *$R_f(2) - 64$ te toplama işleminin birleşme özelliği yoktur:*

  Örneğin MATLAB/Octave ortamında

  $>> 1 - (1/3 + 1/3 + 1/3)$

  ans =

  0

  Fakat

  $>> (1 - 1/3 - 1/3) - 1/3$

  ans =

  $1.1102e-016$ olarak elde edilir.

- *Kayan nokta sayı sisteminde sayılar sıfır noktasının komşuluğunda daha sıkça serpiştirilmiştir ve sonuç olarak işlemlerin gerçekleştirilme sırasına göre farklı yuvarlama hataları oluşabilir ve bu durum işlem sonucunu değiştirebilir:*

  Bu sonucu MATLAB/Octave yazılımlarında kullanılan $eps()$ fonksiyonu yardımıyla gözlemleyebiliriz. Bu fonksiyon $eps(x)$ biçimindeki

kullanımıyla, x sayısı ile bu sayıya en yakın bilgisayar sistemindeki sayı arasındaki uzaklığı verir. Tablo 2.1 de bu sonucu gözlemlemeye çalışıyoruz. Tablonun sol sütununda farklı büyüklükte sayılar yer almakta ve sağ sütununda ise bu sayılar ile bunlara en yakın bilgisayar sayı sistemindeki sayılar arasındaki uzaklık listelenmektedir:

x	$\lvert x - x_f \rvert$
1×10^{-2}	$1.7347E-018$
1	$2.2204E-016$
1×10^{2}	$1.4211E-014$
1×10^{4}	$1.8190E-012$
1×10^{8}	$1.4901E-008$
1×10^{16}	2
1×10^{32}	$1.8014E+016$

Tablo 2.1: Bilgisayar sayıları ve komşuları arasındaki uzaklık

Tablo 2.1 den x değerleri büyüdüğünde, x ile x e en yakın sayılar arasındaki mesafelerin de arttığını gözlemliyoruz:

- 1 noktası ile 1 e en yakın sayı arasındaki uzaklık $2.2204E-016$ iken
- 1×10^{16} ile $1 \times 10^{16} + 1$ sayısı arasında hiç bir bilgisayar sayısı yoktur, çünkü en yakın sayı iki birim uzaklıktadır.

Sıfır komşuluğundaki sayıların birbirlerine çok yakın olması ve mutlak değerce büyük olan kayan nokta sayıları arasındaki uzaklığın kısmen daha büyük olması sonucu sayıların küçükten büyüğe veya büyükten küçüğe sıralanarak toplanması sonucu farklı sonuçlar elde edilebilmektedir. Örneğin

$$\sum_{i=1}^{N} \frac{1}{i^2} = 1 + 1/2^2 + \cdots + 1/N^2$$

büyükten küçüğe toplamını gözönüne alalım: $N = 1e7$ için

```
top=0;
for i=1:N
   top=top+1/(i*i);
end
```

program parçacığı çalıştırıldığında

$$top = 1.64492406684726$$

elde ederiz.

Şimdi de

$$\sum_{i=1}^{N} \frac{1}{(N-(i-1))^2} = 1/N^2 + 1/(N-1)^2 + \cdots + 1/1^2$$

olarak ifade edilen küçükten büyüğe toplamını hesaplayalım:

```
top=0;
for i=N:-1:1
 top=top+1/(i*i);
end
```

program parçacığı çalıştırıldığında

$$top \ = 1.64492406684823$$

elde ederiz. Sonuçların farklı olduğunu görüyoruz.

$$\sum_{i=1}^{\infty} \frac{1}{i^2} = \pi^2/6 = 1.64493406684823$$

olduğu dikkate alınırsa küçükten büyüğe toplamın doğru sonucu ürettiğini gözlemleriz.

- Büyüklükleri çok farklı sayılarla gerçekleştirilen işlemlerde büyük hatalar oluşabilir. Bu duruma örnek olarak aşağıdaki işlem sonuçlarını inceleyelim:

  Gözlem 2.2. *Sonuç olarak büyüklük mertebeleri çok farklı olan bilgisayar sayıları ile yapılan işlemlerde büyük hatalar oluşabilmektedir ve mümkünse işlemlerde çok küçük ve çok büyük sayıların işlemini gerektiren bu tür sonuçların ortaya çıkmaması için alternatif formülasyonlar uygulanmalıdır.*

İşlem	Yaklaşık hata
$0 + 5E - 324 = 4.940656458412465E - 324$	$6E - 326$
$1E5 \times (1 - 1E - 17) = 1E5$	$1E - 12$
$1E10 \times (1E16 + 1E - 1) = 1E26$	$1E + 09$

Tablo 2.2: Farklı mertebeden büyüklüklere sahip sayılarla işlemler ve oluşan hata

– On tabanlı sisteme benzer olarak iki tabanlı sistemde oluşabilecek en büyük bağıl hatanın üst sınırı ise

$$\epsilon_{\max} = \frac{1}{2} \times 2^{-n} = 2^{-(n+1)}$$

olarak elde edilir.

– Öte yandan bilgisayar epsilonu, 1 sayısının komşuluğundaki bir sayının 1 e yuvarlanması sonucu oluşabilen maximum bağıl hata olarak ta tanımlanır. O halde bilgisayar epsilonu 1 ile 1 e en yakın bilgisayar sayısı arasındaki uzaklığın yarısına eşit olmalıdır.

– MATLAB/Octave ortamında $eps()$ fonksiyonu, x_f ile $\overrightarrow{x}_f$ (bir sonraki komşu kayan nokta) arasındaki maksimum bağıl uzaklığı, yani

$$\left| \frac{\overrightarrow{x}_f - x_f}{x_f} \right|$$

oranını verir. $x_f = 1$ olması durumunda bu uzaklık mutlak uzaklığa dönüşür ve bu durumda yukarıdaki tanımda verilen bilgisayar epsilonu

$$\epsilon_{\max} = \frac{1}{2} \times 2^{-n} = \frac{1}{2} eps(1) \tag{2.7}$$

bağıntısından elde edilebilir. Kullandığımız bilgisayar için

$>>$ eps(1)

ans $=$

$2.2204e - 016 (= 2^{-52})$ dir. O halde

$$\epsilon_{\max} = \frac{1}{2} 2^{-52} = 2^{-53}$$

olarak elde edilir. Bu örnekteki

$$p = n + 1 = 53$$

sayısı kayan nokta sayı sisteminin *hassasiyeti* olarak tanımlanır.

2.4.4 $R_f(10) -> R_f(2)$ dönüşüm kaynaklı hatalar

On tabanlı sistemde sonlu basamakla temsil edilebilen bir sayı ikili sistemde sonlu sayıda basamakla temsil edilemeyebilir. Literatürde genelde 0.1 sayısı üzerinden incelenen bu konu diğer bazı tipik ondalıklı sayılar için de geçerlidir[1]. Aşağıdaki tabloda sonlu basamaklı bazı on tabanlı sayıların ikili sistemdeki gösterimleri sunulmaktadır.

$0.1 \rightarrow$	$0.000\overline{1100}$	$0.6 \rightarrow$	$0.100\overline{1100}$
$0.2 \rightarrow$	$0.00\overline{1100}$	$0.7 \rightarrow$	$0.10\overline{1100}$
$0.3 \rightarrow$	$0.0100\overline{1100}$	$0.8 \rightarrow$	$0.\overline{1100}$
$0.4 \rightarrow$	$0.0\overline{1100}$	$0.9 \rightarrow$	$0.1\overline{1100}$
$0.5 \rightarrow$	0.1		

Tabloda

$$\overline{1100} = 110011001100...$$

gösterimi ile 1100 rakamlar grubunun devrettiği ifade edilmektedir.

Tablodan örneğin $0.2 = 0.00\overline{1100}$ devirli sayısıdır. Ancak 32 bit formatta bu sayının mantisi için 23 bitlik bir alan tahsis edildiği için bu formattaki normalize edilmiş gösterimi

$$0.2 \cong (1.10011001100110011001100)_2 \times 2^{-3}$$

dir. Bu sayının on tabanlı sistemdeki karşılığı ise

$$1.9999996E - 001$$

olarak elde edilir.

Böylece 0.2 sayısının ikili sistemdeki devirli gösterimi, tahsis edilen bellek alanına yerleştirilemeyerek gösterimde hata oluşmasına neden olunmuştur.

O halde yukarıdaki tabloda ikili tabanda devirli gösterime sahip her sayı ile gerçekleştirilen işlemlerde çok özel durumlar bile olsalar, zaman zaman

sorun yaşanma ihtimali vardır. Bu noktayı vurgulamak üzere aşağıdaki örneği inceleyelim:

$>> toplam = 0;\ for\ i = 1 : 10\ toplam = toplam + 0.2; end$

$>> z = (2 - toplam) * 1E16$

z = 2.220446049250313

elde ederiz. Yukarıdaki örnekten on adet 0.2 nin toplamının 2 olmasını bekleriz. Esasen

$>> toplam$

toplam =

2.0000

olduğu görülmektedir, ancak bu sonuç yuvarlatılarak ekrana yansıtılan değerdir, gerçek sonuç 2 ye eşit değildir. Bu durumda çok küçük bir değer olan $(2 - toplam)$ nin $1E16$ gibi çok büyük bir sayı ile çarpımı sonucunda, normalde sıfır olması beklenen z değeri yukarıda belirtildiği gibi hatalı olarak elde edilmiştir.

2.5 Anlamlı basamak kaybı hatası

Yukarıdaki örnekte taban dönüşümü sonucu elde edilen birbirine çok yakın iki sayının farkının hesaplanmasında oluşan hata, anlamlı basamak kaybı sonucunda ortaya çıkmıştır. Öncelikle anlamlı basamak sayısını tanımlayalım:

TANIM 2.6. *On tabanlı sistemde eğer* $|\varepsilon_b(x)| \leq 5 \times 10^{-k}$ *ise* x_f *yaklaşımı* x *gerçek değerini* $k-1$ *anlamlı basamakla temsil etmektedir denir.*

Aşağıdaki örnekleri inceleyelim:

- $x_f = 1.1$ yaklaşımı $x = 1$ değerini 0 anlamlı basamakla temsil etmektedir, çünkü

$$|\varepsilon_b(x)| = \frac{1.1 - 1}{1} = 1 \times 10^{-1} < 5 \times 10^{-1}$$

- $x_f = 1.23$ yaklaşımı $x = 1.2$ değerini 1 anlamlı basamkla temsil etmektedir, çünkü

$$|\varepsilon_b(x)| = \frac{1.23 - 1.2}{1.2} = 2.5 \times 10^{-2} < 5 \times 10^{-2}$$

- $x_f = 1.234$ yaklaşımı $x = 1.232$ değerini 2 anlamlı basamakla temsil etmektedir, çünkü

$$|\varepsilon_b(x)| = \frac{1.234 - 1.232}{1.232} \cong 1.6 \times 10^{-3} \leq 5 \times 10^{-3}$$

Anlamlı basamak kaybı sonucu oluşan hatayı aşağıdaki örnek yardımıyla daha yakından inceleyelim.

ÖRNEK 2.6. *Cebirsel olarak birbirine eşit olan*

$$h(x) = 1000(log(x+1) - log(x))$$

ve

$$g(x) = 1000log((x+1)/x)$$

fonksiyonlarını göz önüne alarak, bu fonksiyonların $x = 100000$ noktasındaki değerlerini noktadan sonra 6 basamağa kadar ve yuvarlama prensibine göre çalışan hesaplayıcıda hesaplayarak sonuçları karşılaştırınız.

Çözüm.

x_f ile x in hesaplayıcıda temsil edilen yaklaşımını gösterelim. Bu durumda

$$\begin{aligned} x &= 100000 \text{ ve} \\ y_1 &= log(x+1) \\ &= 5.00000434292310, \\ y_{1f} &\doteq 5.000004 \end{aligned}$$

tür. y_{1f} değeri y_1 değerini 6 anlamlı basamakla temsil eder, çünkü

$$\begin{aligned} \left|\frac{y_{1f} - y_1}{y_1}\right| &= \left|\frac{5.000004 - 5.00000434292310}{5.00000434292310}\right| \\ &= 6.8585 \times 10^{-8} < 5 \times 10^{-7} \end{aligned}$$

dir. Öteyandan

$$y_2 = y_{2f} = log(x) = 5;$$

dir. O halde

$$h(100000) = 1000(y_{1f} - y_{2f}) = 0.004;$$

ve

$$\begin{aligned} g(100000) &= 1000log((10001)/10000) \\ &\doteq 1000 \times 0.000043 = 0.004343. \end{aligned}$$

olarak elde edilir. Gerçek sonuç ise 0.00434292310448164 dir. y_{1f} değeri 6 anlamlı basamağa sahipken birbirine çok yakın y_{1f} ve y_{2f} sayılarının farkı

$$y_{1f} - y_{2f} = 0.000004$$

olarak elde edilir. $y_{1f} - y_{2f}$ yaklaşımı,

$$y_1 - y_2 = 0.00000434292310469431$$

gerçek değerini *hiçbir anlamlı basamakla temsil etmez*, çünkü

$$\begin{aligned} & \left| \frac{(y_{1f} - y_{2f}) - (y_1 - y_2)}{(y_1 - y_2)} \right| \\ = & \left| \frac{0.000004 - 0.00000434292310469431}{0.00000434292310469431} \right| \\ = & \ 7.896\,13576910079 \times 10^{-2} < 5 \times 10^{-1} \end{aligned}$$

olup, anlamlı basamak sayısı tanımında $k = 1$ dir. Dolayısıyla fark işlemi anlam basamağı kaybına neden olmuştur. Söz konusu basamak kaybı, birbirine eşit olan h ve g fonksiyonları yardımıyla elde edilen sonuçların farklılık göstermesine neden olmuştur. g ile elde edilen sonucun gerçek sonuca çok yakın, diğerinin ise gerçek sonuçtan çok farklı olduğunu gözlemliyoruz.

O halde birbirine yakın değerler alması muhtemel değişkenlerin farklarının alınmasından kaçınmak gerekmektedir. Bunun için yukarıdaki örnekte olduğu üzere, verilen ifade belirtilen değişken değerlerinin farkının hesaplanmasını içermeyen alternatif biçimlerde yazılmalıdır. Örneğin aşağıdaki tabloda sol tarafta yer alan fonksiyon yazılımları yerine belirtilen x komşuluğunda sağ sütunda yer alan denk formülasyonlar kullanılmalıdır.

$f(x)$	$\rightarrow$	$f(x)$
$ln(x) - ln(y)$	$x \simeq y$	$ln(x/y)$
$1 - 1/(1+x)$	$x \simeq 0$	$x/(1+x)$
$1 - \cos(x)$	$x \simeq 0$	$\sin^2(x)/(1+\cos(x))$
1-sin$(x)/x$	$x >> 1$	$x\hat{}2/3! - x\hat{}4/5! + \cdots$

Alıştırmalar 2.1.

1. *Aşağıda verilen saylar için*

   (a) *kesme prensibine göre kayan noktadan sonra* 4 *basamaklı ve*

   (b) *yuvarlama prensibine göre de* 4 *basamaklı yaklaşımların kullanılması durumunda oluşacak olan mutlak ve bağıl hataları hesaplayınız.*

   - $x = 3.14159265358979$
   - $x = 2.71828182845905$
   - $x = 1.41421356237310$
   - $x = 1.73205080756888$

2. $$x_f = \pm q_f \times 10^e,\ q_f = d_0.d_1d_2,\ d_{0,1,2} = 0, 1, \ldots, 9\ ; d_0 \neq 0$$

   ve

   $$e = -1, 0, 1$$

   üs değerlerine sahip bir R_f *sayı sistemi gözönüne alalım ve gerçekleştirilen her işlem sonucunun* R_f *de temsil edilebilen aralıkta olup, ancak* R_f *de yer almaması durumunda, sistemde bulunan en yakın sayıya yuvarlatıldığını kabul edelim. Buna göre*

   - R_f *in elemanları(sayıları) nelerdir?*
   - R_f *in en büyük pozitif ve en küçük pozitif sayıları nelerdir?*
   - R_f *kümesinin toplama işlemine göre kapalılık özelliğini sağlamadığını bir örnekle gösteriniz.*
   - R_f *de toplama işlemine göre birleşme özelliği olmadığını bir örnekle gösteriniz.*
   - R_f *deki bağıl yuvarlama hatalarının üst sınırını temsil eden bilgisayar epsilonu nedir?*

- R_f *de inf ve NaN gösterimlerine sahip olabilecek birer işlem tanımlayınız*
- R_f *deki subnormal sayılar kümesini belirleyiniz.*

3. *Aşağıda verilen on tabanlı sayıların karşılarında verilen iki tabanlı gösterimlere sahip olduklarını gösteriniz*

$3 \rightarrow$	$(11)_2$	$50 \rightarrow$	$(110010)_2$
$8 \rightarrow$	$(1000)_2$	$100 \rightarrow$	
$10 \rightarrow$	$(1010)_2$	$200 \rightarrow$	$(11001000)_2$
$20 \rightarrow$	$(10100)_2$	$500 \rightarrow$	$(111110100)_2$
$40 \rightarrow$	$(101000)_2$	$1000 \rightarrow$	$(1111101000)_2$

4. *Aşağıda verilen on tabanlı sayıların iki tabanlı gösterimlerini belirleyiniz.*

- 25.1
- 33.25
- 75.454
- 96.2456

5. *Soru 3 de elde ettiğiniz iki tabanlı gösterimleri on tabanlı sisteme dönüştürerek, başlangıçtaki on tabanlı sayıları elde ettiğinizi kontrol ediniz.*

6. *Aşağıda verilen iki tabanlı sayıların on tabanlı gösterimlerini belirleyiniz*

- $(11001)_2$
- $(1001011)_2$
- $(1100000.101)_2$
- $(100001.1101)_2$

7. *Soru 6 da verilen iki tabanlı sayıların* $R_f(2) - 32$ *bit bellek gösterimlerini belirleyiniz.*

8. *Soru 6 da verilen iki tabanlı sayıların* $R_f(2) - 64$ *bit bellek gösterimlerini belirleyiniz.*

9. *Aşağıda verilen kesirli sayıların iki tabanlı gösterimlerinin doğruluğunu kontrol ediniz.*

   - $2.5 \rightarrow (10.1)_2$
   - $1.25 \rightarrow (1.01)_2$
   - $1.125 \rightarrow (1.001)_2$
   - $1.3125 \rightarrow (1.0101)_2$

10. *Aşağıda verilen kesirli sayıların iki tabanlı gösterimlerinin doğruluğunu kontrol ediniz.*$\overline{1001}$ *üs çizgi notasyonu sayının devirli sayı olduğunu ifade etmektedir.*

    - $0.1 \rightarrow (0.0\overline{0011})_2$
    - $3.6 \rightarrow (1.1\overline{1001})_2$

11. *Bilgisayar sisteminizde* $[1, 2)$ *aralığındaki kayan nokta sayıları arasındaki uzaklığın birbirine eşit olduğunu MATLAB/Octave* eps *komutu yardımıyla gözlemleyiniz. Buna göre örneğin* $eps(1) = eps(1.5) = eps(1.8)$ *olmalıdır.*

12. *Bilgisayar sisteminizde* $[2^n, 2^{n+1})$ *aralığında* $(\mathsf{n} = 0, 1, 2, ...)$ *yer alan sayılar arasındaki uzaklıkların birbirine eşit olduğunu ve her bir aralıktaki sayılar arasındaki mesafenin bir önceki aralıktaki sayılar arasındaki mesafenin* 2 *katı olduğunu gözlemleyiniz.*

13. *Bilgisayar sisteminizdeki pozitif ve en küçük subnormal sayıyı belirleyiniz. Elde ettiğiniz sayıdan daha küçük ve negatif olmayan tek sayı sıfır olmalıdır.*

14. *Bilgisayarınızda temsil edilebilecek en büyük sayının 64-bit formatta* $(2 - eps) \times 2^{1023}$ *olduğunu MATLAB/Octave ortamında gözlemleyiniz. Bu durumda* 2^{1024} *sayı sisteminizde yer alır mı?*

15. *Aşağıda verilen yaklaşımların karşılarında verilen değerleri kaç anlamlı basamakla temsil ettiğini açıklayınız*

    - $x_f = 12.36$, $x = 12.345$
    - $x_f = 0.0013$, $x = 0.00125$
    - $x_f = 0.000011$, $x = 0.000012$

- $x_f = 3.14159, x = 3.141592653589793$

16. *Anlam basamak kaybı hatasının önlenmesi için aşağıdaki fonksiyonlar belirtilen nokta komşuluğunda alternatif olarak nasıl yazılabilir?*

    - $e^{-x}, x > 0$
    - $(-b + \sqrt{b^2 - 4c})/2, b >> 0, c \simeq 0$(*burada* $>>$ *sembolü çok büyük anlamındadır,* $a \simeq b$ *gösterimi ise* a *nın* b *ye çok yakın olduğunu ifade etmektedir).*
    - $e^{x-y}, x \simeq y$
    - $\sqrt{x+1} - \sqrt{x}, x >> 1$

17. *Aşağıda verilen Program 2.2 yi MATLAB veya OCTAVE ortamında çalıştırarak, yukarıda elde ettiğiniz taban dönüşümlerinin doğruluğunu kontrol ediniz.*

```
Test
>> onluikili(24.5)
İkili sayi
1 1 0 0 0.1
Onlu sayi
2.450000e+001
>> onluikili(0.1)
İkili sayi
0.0 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1
Onlu sayi
9.999996e-002
```

```
% On tabanlı kesirli pozitif sayıyı iki tabanlı kesirli
% sayıya dönüştürür ve elde edilen iki tabanlı sayıyı tekrar
% on tabanlı sisteme dönüştürerek sonucu kontrol eder.
% Mart 2009, ec
%-------------------------------------------------------------
function onluikili(sayi)
 tamsayi=fix(sayi);tamsayim=tamsayi;kesirsayi=abs(sayi-tamsayi);
 i=1;
 if tamsayim>0
   while tamsayi>0
     k(i)=mod(tamsayi,2);      % k nın elemanları girilen
     tamsayi=floor(tamsayi/2); % sayının tam kısmının
     i=i+1;                    % ikili sistemdeki karşılığıdır.
   end
 else
   k=0;
 end

 x0=2;
 tamon=horner(k,x0); %Horner yöntemi Bölüm 3
                       de incelenmektedir.
 i=1;
 kesir(1)=abs(2*kesirsayi-fix(2*kesirsayi));
 d(1)=fix(2*kesirsayi);
 while kesir(i)>0 &i<24
    i=i+1;
    d(i)=fix(2*kesir(i-1)) ;
    kesir(i)=abs(2*kesir(i-1)-fix(2*kesir(i-1)));
 end
 if kesirsayi<0 d=0;
 end
 x0=1/2;
 tamkesir=horner(d,x0);
 k=fliplr(k);fprintf('ikili sayi\ n');
 sayi=strcat(num2str(k),'.'); sayi=strcat(sayi,num2str(d));
 fprintf('%5s',sayi); fprintf('\ n');
 fprintf('Onlu sayi\ n'); fprintf('%d \ n',tamon+tamkesir);
 function tamon=horner(k,x0)
 n=length(k)-1;b=zeros(n+1);b(n+1)=k(n+1);
 for i=n:-1:1
   b(i)=k(i)+x0*b(i+1);
 end
 if x0==2 tamon=b(1); else tamon=0.5*b(1);end
```


Program 2.2: Kesirli sayı dönüşüm uygulaması

Kaynaklar

[1] Atkinson, K. An Introduction to Numerical Analysis, John Wiley & Sons, 1988.

[2] Coşkun, E. OCTAVE ile Sayısal Hesaplama ve Kodlama(URL:aves.ktu.edu.tr/erhan/dokumanlar).

[3] Kincaid, D., Cheney, W., Numerical Analysis, Brooks/Cole, 1991.

[4] Mathews, J., Numerical Methods for Mathematics, Science and Engineering, Prentice-Hall, 1997.

[5] MATLAB, Mathworks(*URL:mathworks.com).*

[6] OCTAVE, GNU özgür yazılım(*URL:OCTAVE.sourceforge.net*).

[7] Overton, M. L., Numerical Computing with IEEE Floating Point Arithmetics, SIAM, New York, 2001.

[8] M., Cleve, Floating point Numbers, (URL:blogs.mathworks.com/cleve/2014/07/07/floating-point-numbers).

[9] Press, H. W. ve ark., Numerical Recipes in C, Cambridge University Press, 1988.

[10] Stoer, J., Bulirsh, R., Introduction to Numerical Analsis, Springer-Verlag, 1976.

Bölüm 3

Taylor Serisi, Polinomu ve Polinomla Yaklaşım Hatası

Bu bölümde

- verilen bir a noktası komşuluğunda yakınsak bir kuvvet seri açılımı ile tanımlanan f fonksiyonu ve serinin sonlu sayıda teriminden oluşan Taylor polinomunu tanıtarak,
- söz konusu *Taylor* polinomunun a noktasının hangi komşuluğunda ilgili f fonksiyonuna yaklaşım için kullanılabileceğini,
- sonlu terimli *Taylor* yaklaşımı ile oluşan sonlandırma(truncation) hatasının verilen bir $\epsilon > 0$ dan küçük olması için kullanılması gereken yaklaşım polinomunun derecesinin nasıl tahmin edilebileceğini,
- bilinen *Taylor* polinomu yardımıyla, verilen benzer fonksiyonlara ait *Taylor* polinomlarının nasıl türetilebileceğini,
- *Taylor* polinomu ile bir fonksiyona yaklaşımın neden gerekli olduğunu,
- Taylor polinomunun herhangi bir nokta kümesi üzerindeki değerlerinin Horner yöntemi ile ve eş zamanlı olarak MATLAB/Octave'ın **vektör cebirsel** işlem yeteneği yardımıyla nasıl hesaplanabileceğini,
- iki değişkenli fonksiyonların *Taylor* polinomlanarının nasıl hesaplandığını ve

- verilen fonksiyonun, verilen bir nokta komşuluğundaki birinci dereceden *Taylor* polinom yaklaşımı yardımıyla, bilgisayar ortamında yürütülen aritmetik işlemlerde oluşan yuvarlama hatalarının nasıl birikebileceğini inceliyoruz.

Bu bölümde yer veremediğimiz detaylar için [1],[2] [6] ve [7] nolu temel referans kaynaklarını öneriyoruz.

3.1 *Taylor* serisi ve polinomu

a, $c_n \in R$, $n = 0, 1, \cdots$ sabitleri ve keyfi $x \in R$ için

$$\sum_{n=0}^{\infty} c_n (x-a)^n := c_0 + c_1(x-a) + \cdots + c_n(x-a)^n + \cdots \tag{3.1}$$

toplamına a merkezli ve sabit katsayılı bir kuvvet serisi adı verildiğini hatırlayalım. Eğer (3.1) serisi $R > 0$ sabit olmak üzere, yalnız $|x-a| < R$ aralığındaki x ler için sonlu değerlere sahipse, seriye $(a-R, a+R)$ aralığında *yakınsaktır* denir ve bu aralığa serinin yakınsaklık aralığı ve R ye de yakınsaklık yarıçapı adı verilir. Bu aralığın dışındaki noktalar veya aralıklarda (3.1) toplamı sonlu bir değere sahip olmadığı için seriye bu tür nokta veya aralıklarda *ıraksak* seri adı verilir.

Yakınsaklık aralığı içerisinde kuvvet serisi bir fonksiyon tanımlar: Her bir noktadaki değeri (3.1) toplamına eşit olan f fonksiyonu

$$f(x) := \sum_{n=0}^{\infty} c_n (x-a)^n, x \in (a-L, a+L) \tag{3.2}$$

olarak tanımlanır. Bu durumda (3.2) in sağ tarafındaki seriye, f nin a noktası komşuluğundaki kuvvet serisi veya *Taylor serisi* adı verilir.

Hatırlatma 3.1. *Kuvvet serileri yakınsaklık bölgeleri içerisinde terim terime türevlenebilir ve interallenebilirler. Türev ve integral işlemleri sonucunda elde edilen seriler de* aynı aralıkta *yakınsaktırlar.*

Serinin c_n katsayıları ile f nin ve türevlerinin a noktasındaki değerleri arasında aşağıda türetilen bir ilişki mevcuttur:

$$f(x) = c_0 + c_1(x-a) + \cdots + c_n(x-a)^n + \cdots$$

ifadesinden yakınsaklı aralığı içerisinde terim terime türev alınabileceği kuralını kullanarak,

$$\begin{aligned} f(a) &= c_0, \\ f'(a) &= c_1, \\ f''(a) &= 2c_2 \\ &\cdots \\ f^{(n)}(a) &= n!c_n \end{aligned}$$

olduğu kolayca görülür. O halde f fonksiyonunun a noktası komşuluğundaki $n-$inci dereceden *Taylor polinomu*

$$\begin{aligned} P_n(x) \quad : \quad &= f(a) + f'(a)(x-a) + \frac{1}{2!}f''(a)(x-a)^2 + \cdots + \frac{1}{n!}f^{(n)}(a)(x-a)^n \\ &= \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!}(x-a)^k \end{aligned}$$

olarak tanımlanır ve yakınsaklık aralığı içerisindeki her x noktasında

$$f(x) := \lim_{n\to\infty} P_n(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!}(x-a)^k \tag{3.3}$$

olarak ifade edilebilir.

Uyarı. *Herhangi bir f fonksiyonunun bir a noktası komşuluğunda yakınsak Taylor serisine sahip olması için a noktasında fonksiyonun bütün basamaktan türevlerinin mevcut olması ve ayrıca oluşturulan kuvvet serisinin a noktası komşuluğundaki x noktaları için $f(x)$ değerine yakınsak olması gerekir.*

TANIM 3.1. *Bir a noktası komşuluğunda (3.3) ile tanımlanan yakınsak kuvvet seri açılımına sahip fonksiyona a noktasında analitik fonksiyon adı verilmektedir.*

Buna göre bir noktada sürekli olmayan veya herhangi bir basamaktan türevi olmayan fonksiyonun ilgili nokta komşuluğundaki *Taylor* seri açılımından bahsedemeyiz. Öte yandan çok özel durumlar olmasına rağmen, bir fonksiyonun bir noktada her basamaktan türevin mevcut olması da fonksiyonun

o nokta komşuluğunda yakınsak kuvvet seri açılımına sahip olmasını garanti etmez(Alıştırma 9).

ÖRNEK 3.1. *$f(x) = cos(x)$ fonksiyonunun $a = 0$ noktası komşuluğundaki Taylor serisini ve serinin kısmi toplamlar dizisini belirleyiniz.*

Çözüm.

f nin $a = 0$ noktasında sürekli ve her basamaktan türevinin mevcut olduğunu biliyoruz. Ayrıca

$$\begin{aligned} f(0) &= 1, f'(x) = -\sin(x), f'(0) = 0, f''(x) = -\cos(x), f''(0) = -1, \\ f'''(x) &= \sin(x), f'''(0) = 0, f^{(4)}(x) = \cos(x), f^{(4)}(0) = 1, ... \end{aligned}$$

değerlerini elde ederiz. O halde (3.3) den

$$\begin{aligned} \cos(x) &= f(0) + f'(0)x + \frac{1}{2!}f''(0)x^2 + \cdots + \frac{1}{n!}f^{(n)}(0)x^n + \cdots \\ &= 1 - \frac{1}{2!}x^2 + \frac{1}{4!}x^4 - ... \\ &= \sum_{k=0}^{\infty} \frac{(-1)^k}{(2k)!}x^{2k} \end{aligned}$$

ile ifade edilen *Taylor* seri açılımını belirleriz. Buna göre serinin kısmi toplamlar dizisi

$$\begin{aligned} P_0(x) &= P_1(x) = 1 \\ P_2(x) &= P_3(x) = 1 - \frac{1}{2!}x^2 \\ P_4(x) &= P_5(x) = 1 - \frac{1}{2!}x^2 + \frac{1}{4!}x^4 \\ &\vdots \\ P_{2n}(x) &= P_{2n+1}(x) = 1 - \frac{1}{2!}x^2 + \frac{1}{4!}x^4 - ... + \frac{(-1)^n}{(2n)!}x^{2n}, \\ &\vdots \end{aligned}$$

olarak ifade edilir.

$P_n(x)$ Taylor polinomu f fonksiyonunun a noktası komşuluğundaki kuvvet serisinin kısmi toplamlar dizisinin bir elemanıdır.

Taylor teoremi olarak bilinen aşağıdaki sonuç, $P_n(x)$ polinomunu yukarıda elde edilen yöntemden daha farklı bir yöntemle türetir ve $f(x)$ yerine $P_n(x)$ polinomunun kullanılması durumunda oluşacak olan hata için birbirine denk olan iki formülasyon önerir.

TEOREM 3.1. *(Taylor teoremi) $f \in C^{n+1}[a,b]$, ve $x, x_0 \in [a,b]$ seçilsin. Bu taktirde*

$$f(x) = f(x_0)+(x-x_0)f'(x_0)+\frac{(x-x_0)^2}{2!}f''(x_0)+\cdots+\frac{(x-x_0)^n}{n!}f^{(n)}(x_0)+R_n(x) \tag{3.4}$$

olarak ifade edilir. Burada

$$R_n(x) = \frac{1}{n!}\int_{x_0}^{x} f^{(n+1)}(t)(t-x_0)^n dt$$

kalan terimdir veya alternatif olarak

$$R_n(x) = (x-x_0)^{n+1}/(n+1)!f^{(n+1)}(c_x),$$

biçiminde de yazılabilir ve c_x ise x_0 ile x arasında bir noktadır.

İspat. Analizin temel teoreminden

$$f(x) = f(x_0) + \int_{x_0}^{x} f'(t)dt \tag{3.5}$$

olarak yazılır. Teoremin ispatı $\int_{x_0}^{x} f'(t)dt$ integraline kısmi integrasyon yönteminin uygulanmasını esas almaktadır. Buna göre

$$u = f'(t), du = f''(t)dt$$

ile

$$dv = dt$$

den $v = t$ yerine integral sabiti olarak $-x$ seçimiyle $v = t-x$ almak suretiyle

$$\int_{x_0}^{x} f'(t)dt = (t-x)f'(t)|_{x_0}^{x} - \int_{x_0}^{x}(t-x)f''(t)dt \tag{3.6}$$

$$= (x-x_0)f'(x_0) + \int_{x_0}^{x}(x-t)f''(t)dt \tag{3.7}$$

elde ederiz. O halde (3.6) deki ifadeyi (3.5) te yerine yazarak

$$f(x) = f(x_0) + (x - x_0)f'(x_0) + \int_{x_0}^{x} (x - t)f''(t)dt \tag{3.8}$$

elde ederiz. (3.8) daki integral için de

$$u = f''(t), dv = (x - t)dt$$

ve

$$du = f'''(t)dt, v = -(x - t)^2/2$$

dönüşümleri ile

$$\begin{aligned} \int_{x_0}^{x} (x - t)f''(t)dt &= -f''(t)(x - t)^2/2|_{x_0}^{x} + \frac{1}{2!}\int_{x_0}^{x} f'''(t)(x - t)^2 dt \\ &= \frac{(x - x_0)^2}{2!}f''(x_0) + \frac{1}{2!}\int_{x_0}^{x} f'''(t)(x - t)^2 dt \end{aligned} \tag{3.9}$$

elde ederiz. (3.9) ifadesini (3.8) da yazarak

$$f(x) = f(x_0)+(x-x_0)f'(x_0)+\frac{(x - x_0)^2}{2!}f''(x_0)+\frac{1}{2!}\int_{x_0}^{x} f'''(t)(x-t)^2 dt \tag{3.10}$$

olarak bir adım daha aranan gösterime yaklaşırız. Tümevarım adımı gereği $n - 1$ için

$$\begin{aligned} f(x) &= f(x_0) + (x - x_0)f'(x_0) + \frac{(x - x_0)^2}{2!}f''(x_0) + \\ &\quad \cdots + \frac{1}{(n-1)!}\int_{x_0}^{x} f^{(n)}(t)(x - t)^{n-1} dt \end{aligned} \tag{3.11}$$

olduğunu kabul ederek,

$$u = f^{(n)}(t), dv = (x - t)^{n-1}dt$$

ve

$$du = f^{(n+1)}(t)dt, v = -(x - t)^n/n$$

dönüşümleri ile elde edilen

$$\int_{x_0}^{x} f^{(n)}(t)(x-t)^{n-1}dt = -\frac{1}{n}f^{(n)}(t)(x-t)^n|_{x_0}^{x} + \frac{1}{n}\int_{x_0}^{x} f^{(n+1)}(t)(x-t)^n dt$$

ifadesini (3.11) de yazarak aranan sonucu elde ederiz.

Öte yandan c_x, x_0 ile x arasında bir nokta olmak üzere integraller için ortalama değer teoreminin bir sonucu olarak

$$R_n(x) = \frac{1}{n!}\int_{x_0}^{x} f^{(n+1)}(t)(t-x_0)^n dt = (x-x_0)^{n+1}/(n+1)!f^{(n+1)}(c_x) \quad (3.12)$$

elde edilir.

3.2 *Taylor* serisi ve yakınsaklık bölgesi

$a = 0$ olması durumda (3.4)gösterimine f fonksiyonunun *Maclaurin*[1] açılımı adı verilmektedir. Elemanter bazı fonksiyonların Maclaurin açılımları ve yakınsaklık bölgeleri aşağıda verilmektedir:

$$\begin{aligned} sin(x) &= \sum_{n=0}^{\infty}(-1)^n x^{2n+1}/(2n+1)! \qquad (3.13)\\ &= x - x^3/3! + x^5/5! - \ldots + (-1)^n x^{(2n+1)}/(2n+1)! + \cdots, \end{aligned}$$

Oran testi adı verilen test yardımıyla

$$\sum_{n=0}^{\infty} c_n(x-a)^n$$

kuvvet serisi

$$lim_{n\to\infty}\left|\frac{c_{n+1}(x-a)^{n+1}}{c_n(x-a)^n}\right| = |x-a|lim_{n\to\infty}\left|\frac{c_{n+1}}{c_n}\right| < 1$$

veya

$$L = lim_{n\to\infty}\left|\frac{c_n}{c_{n+1}}\right|$$

[1]Colin Maclaurin, 1698-1746, İskoçyalı matematikçi

olmak üzere

$$|x-a| < L$$

eşitsizliğini sağlayan $(a-L, a+L)$ aralığında yakınsaktır. O halde $sin(x)$ fonksiyonu için

$$\begin{aligned} L &= lim_{n\to\infty}\left|\frac{c_n}{c_{n+1}}\right| \\ &= lim_{n\to\infty}\left|\frac{1}{(2n+1)!}\frac{(2(n+1)+1)!}{1}\right| \\ &= lim_{n\to\infty}(2n+2)(2n+3) = \infty \end{aligned}$$

olur, yani yukarıda (3.13) te verilen tanımlanan kuvvet serisi $sin(x)$ fonksiyonunu $(-\infty, \infty)$ aralığında temsil eder.

Benzer biçimde

$$\begin{aligned} cos(x) &= \sum_{n=0}^{\infty}(-1)^n x^{2n}/(2n)! \\ &= 1 - x^2/2! + x^4/4! - \ldots + (-1)^n x^{(2n)}/(2n)! + \cdots \end{aligned} \tag{3.14}$$

seri gösterimi $cos(x)$ fonksiyonunu $(-\infty, \infty)$ aralığında temsil eder.

Uyarı. *Kuvvet serileri, temsil ettikleri fonksiyonları bu fonksiyonların tanım kümelerinde değil, sadece ilgili serilerin yakınsak oldukları bölgelerde temsil ederler.*

Örneğin,

$$\begin{aligned} ln(x+1) &= \sum_{n=1}^{\infty}(-1)^{(n+1)} x^n/n \\ &= (x - x^2/2 + x^3/3 - \ldots + (-1)^{(n+1)} x^n/n + \cdots, x \in (-1, 1] \end{aligned} \tag{3.15}$$

açılımı için oran testi $|x| < 1$ için yakınsaklığı garanti eder. $x = 1$ için de elde edilen serinin yakınsaklığı alterne sayı serileri için yakınsaklık kriteri yardımıyla kolayca görülebilir. O halde yukarıdaki açılım $ln(x+1)$ fonksiyonunu sadece $(-1, 1]$ aralığında temsil eder. Örneğin $x = 2$ için $ln(x+1) = ln(3)$ tanımlı iken ilgili seri bu noktada sonlu bir değere sahip değildir.

Benzer problem

$$1/(1-x) = \sum_{n=0}^{\infty} x^n = 1 + x + x^2 + \cdots + x^n + \cdots \tag{3.16}$$

açılımı için de geçerlidir. Seri açılımı ve sol tarafındaki fonksiyon sadece $(-1, 1)$ aralığında birbirine eşittir. Örneğin bu aralığın dışındaki $x = 2$ noktası için $1/(1-x) = 1/(-1) = -1$ iken, sağ taraftaki toplam bu noktada sonlu bir değere sahip değildir.

Ancak

$$e^x = \sum_{n=0}^{\infty} x^n/n! = 1 + x/1! + x^2/2! + \cdots + x^n/n! + \cdots \tag{3.17}$$

açılımındaki e^x fonksiyonu ve sağ tarafındaki sonsuz toplam her $x \in (-\infty, \infty)$ için aynı değere sahiptir, yani fonksiyon ve seri reel sayılar kümesi üzerinde birbirine eşittirler.

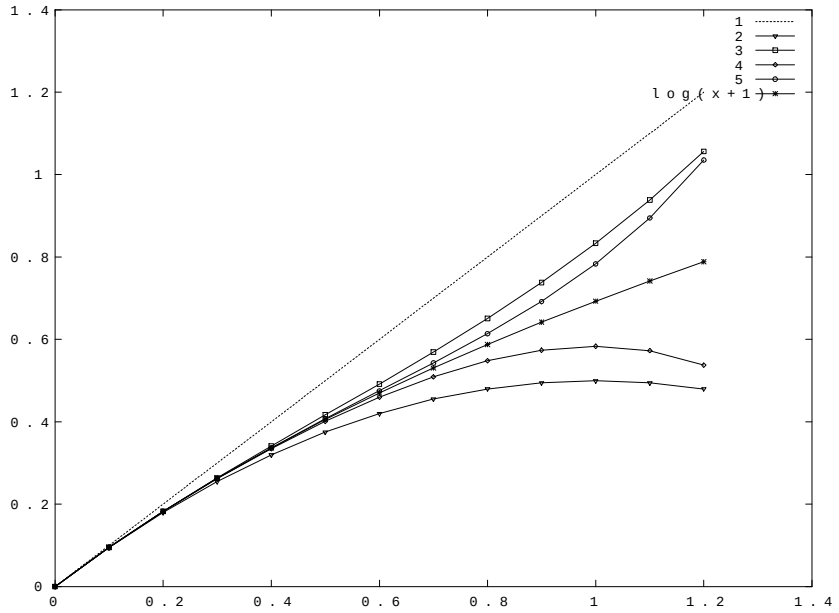
3.3 Yakınsaklık bölgesinde *Taylor* polinomları ile yaklaşım

Bir fonksiyona yaklaşım için *Taylor* polinomu kullanılırken, yaklaşımın ilgili *Taylor* serisinin yakınsaklık bölgesinde geçerli olduğunu her zaman göz önünde bulundurmak gerekir. Aksi taktirde hangi dereceden polinom kullanılırsa kullanılsın, elde edilen yaklaşımlar olumlu sonuçlar vermezler. Bu durumu aşağıdaki örnekle inceleyelim:

ÖRNEK 3.2. *$f(x) = ln(x+1)$ fonksiyonunun $a = 0$ noktası komşuluğunda ve $[0, 1.2]$ aralığında Taylor polinomlarını belirleyerek belirtilen aralıkta artan n değerlerine rağmen yakınsamanın gerçekleşmediğini gözlemleyiniz.*

Çözüm.

$f(x) = ln(x+1)$ fonksiyonunun $a = 0$ noktası komşuluğunda (3.15) ile verilen *Taylor* serisi $(-1, 1]$ aralığında yakınsaktır. Şekil 3.1 de $[0, 1.2]$ aralığında $n = 2, 3, \ldots, 6$ için $P_n(x)$ yaklaşımları(noktalı) ve f fonksiyonunun grafiği(çizgi) gösterilmektedir.



Şekil 3.1: $[0, 1.2]$ aralığında $\ln(x+1)$ ve ilk beş Taylor yaklaşımının grafiği

n nin tek değerleri için elde edilen yaklaşımlar $x = 1.2$ noktasında f nin grafiğinin üst kısmında yer alırken, çift değerler için elde edilen yaklaşımlar ise grafikleri f nin grafiğinin aşağısında yer alan yaklaşımlardır. Artan n değerleri için elde edilen yaklaşımların $x = 1.2$ noktasında f nin grafiğinden gittikçe uzaklaştıkları görülmektedir. Bu durum, Tablo 3.1 de verilen yaklaşım hataları için sonsuz normlarından da açıkça görülmektedir.

n	$\|f(x) - P_n(x)\|_\infty$
2	0.30846
4	0.25086
6	0.25086
8	0.27645
10	0.32232
15	0.54321

Tablo 3.1: Yaklaşım hataları

Tablo 3.1 den *Taylor* polinomları ile elde edilen yaklaşımların, *Taylor* serisinin yakınsaklık aralığı içerisinde yer almayan x noktaları için iyi sonuç

vermeyeceğini gözlemliyoruz.

Uyarı. *Bir Taylor polinomunu bir nokta komşuluğunda ilgili fonksiyona yaklaşım amacıyla kullanmadan önce, polinomun ait olduğu Taylor serisinin yakınsaklık bölgesine dikkat edilmelidir.*

3.4 Uygun dereceden *Taylor* yaklaşım polinomu

Bazı uygulamalarda belirtilen yakınsaklık aralığı içerisinde verilen bir maksimum hata ile yaklaşım sağlayan *Taylor* polinomunun derecesinin de tahmin edilebilmesi gerekmektedir. Bu bağlamda (3.12) ile verilen hata tahmin formülünden faydalanabiliriz.

ÖRNEK 3.3. *$f(x) = ln(x+1)$ fonsiyonu için $a = 0$ noktası komşuluğunda ve $[0,1]$ aralığında $\epsilon = 0.1$ den küçük sonsuz normu hatası ile elde edilen Taylor polinomunun derecesini belirleyiniz.*

Çözüm.

Öncelikle

$$f^{(n+1)}(x) = (-1)^n n!/(1+x)^{(n+1)}$$

olarak elde edildiğine dikkat edelim. O halde herhangi $c_x \in (0,1)$ için

$$\begin{aligned} \|f - P_n\|_\infty &= max_{0\le x\le 1} \left| \frac{x^{(n+1)}}{(n+1)!} f^{(n+1)}(c_x) \right| \\ &= max_{0\le x\le 1} \left| \frac{x^{(n+1)}(-1)^n n!}{(n+1)!(1+c_x)^{(n+1)})} \right| \\ &\le 1/(n+1) < \epsilon = 0.1 \end{aligned}$$

için, $n \ge 10$ olması gerektiği tahmin edilir. Ancak bu tahmin aşırı temkinli bir tahmindir ve gerçekte

$$\|f - P_5\|_\infty = 0.0902 < 0.1$$

olup $n \ge 5$ olması yeterlidir.

3.5 *Taylor* açılımı bilinen bir fonksiyon yardımıyla benzer fonksiyonların açılımları

Taylor açılımı bilinen bir fonksiyon yardımıyla benzer fonksiyonların açılımları hesaplanabilir.

ÖRNEK 3.4. *e^{-x^2} fonksiyonunun $x = 0$ noktası komşuluğundaki Taylor açılımını e^x in Taylor açılımı cinsinden hesaplayınız. Farklı n değerleri için $\|e^{-x^2} - P_n(x)\|_\infty$ normlarını hesaplayarak hatanın artan n değerleri için sıfıra nasıl yakınsadığını gözlemleyiniz?*

Çözüm.

Öncelikle e^{-x} in açılımını e^x in *Taylor* açılımında x yerine $-x$ yazarak elde edebileceğimize dikkat edelim:

$$e^{-x} = 1 - x/1! + x^2/2! - \ldots + (-1)^n x^n/n! + \cdots$$

ve dolayısıyla x yerine x^2 yazmak suretiyle

$$e^{-x^2} = 1 - x^2/1! + x^4/2! - \ldots + (-1)^n x^n/n! + \cdots$$

elde ederiz. (3.17) serisinin yakınsaklık yarıçapının sonsuz olduğuna dikkat edelim.

$[-2, 2]$ aralığında hesaplanan

$$\|e^{-x^2} - P_n(x)\|_\infty$$

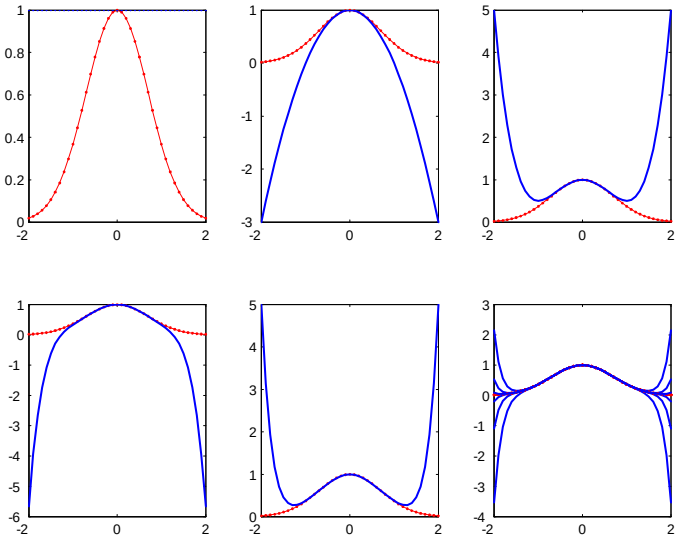
hatalar farklı n değerleri için aşağıdaki tabloda verilmektedir.

n	0	2	4	6	8	10	12	16	20
$\|\|f(x) - P_n(x)\|\|_\infty$	0.98	3.02	4.98	5.68	4.98	3.55	2.14	0.51	0.08

Tablodan hatanın n yaklaşım derecesinin fonksiyonu olarak monoton biçimde azalmadığını görüyoruz. Diğer bir değimle P_{n+1} yaklaşımının P_n den daha iyi olması gerekmemektedir. Ancak ilgili *Taylor* serisinin yakınsaklık aralığı içerisinde

$$||f(x) - P_n(x)||_\infty \to 0$$

olduğunu örnek üzerinden de gözlemliyoruz. e^{-x^2} ve ilgili $P_n(x)$ *Taylor* polinomlarının grafikleri ise Şekil (3.2) de verilmektedir.



Şekil 3.2: e^{-x^2} ve Taylor yaklaşımları

Benzer biçimde $sinx$ fonksiyonunun *Taylor* açılımını kullanarak $x \neq 0$ için

$$sin(x)/x = 1 - x^2/3! + x^4/5! - x^6/7! + x^8/9! + \cdots$$

elde edebiliriz.

3.6 Neden *Taylor* polinomları?

Bilinen analitik yöntemlerle bir a noktasının komşuluğunda yakınsak bir kuvvet serisi ile ifade edilebilen bir fonksiyonla çalışmanın zor veya mümkün olmadığı durumlarda bu noktanın komşuluğunda fonksiyonu temsilen kuvvet serisinin açılımın belirli sayıda teriminden oluşan *Taylor* yaklaşımı kullanılabilir. Örneğin analitik yöntemlerle hesaplanamayan

$$\int_{-1}^{1} e^{-x^2} dx$$

integrali için bir sayısal yaklaşımı *MATLAB quadl* fonksiyonu yardımıyla **1.493648265**624569 olarak elde ederiz. e^{-x^2} için sıfır noktası komşuluğunda

yaklaşım olarak P_{20} *Taylor* polinomunun integralini hesaplamak suretiyle ise **1.49364826**7647435 elde ederiz. Sonuçların virgülden sonra *sekiz* basamağa kadar aynı olduklarını gözlemliyoruz.

- a noktasını içeren yeterince küçük bir $[a-L, a+L]$ *Taylor* serisi yakınsaklık aralığında

$$\int_{a-L}^{a+L} f(x)dx \cong \int_{a-L}^{a+L} P_n(x)dx$$

  yaklaşımı alınabilir. Ancak integral aralığının büyük olması durumunda daha yüksek dereceden polinomun kullanılması gerekeceği için *Taylor* polinomu yerine daha uygun polinomlar kullanılmalıdır. Bu konuyu sayısal integrasyon yöntemleri bölümünde inceleyeceğiz.

- Verilen bir fonksiyonun uygun bir a noktası komşuluğundaki birinci dereceden *Taylor* polinomu, a noktası komşuluğunda fonksiyon için belki de en fazla kullanılan bir yaklaşımdır:

  - Bölüm 6 da inceleyeceğimiz fonksiyon sıfıryeri belirleme problemlerinde güncel olarak kullanılan *Newton* yöntemi, her noktada fonksiyonun birinci dereceden *Taylor* polinomu yaklaşımını kullanır ve bu polinomun sıfıryerini, kendi sıfıryeri için bir yaklaşım olarak kabul eder.
  - Bazı sayısal türev yöntemleri, fonksiyon türevi için ilgili Taylor polinomunun türevini yaklaşım olarak kabul eder.

- Diferensiyel denklemler için sayısal yöntemlerin hata analizinin gerçekleştirilmesinde *Taylor* yaklaşımları sıkça kullanılır.

3.7 *Horner* yöntemi ile polinom değer veya değerlerinin hesabı

$$P_n(x) = a_1x^n + a_2x^{n-1} + \cdots + a_nx + a_{n+1}$$

olarak ifade edilen polinomun x_0 noktasındaki değeri

$$P_n(x_0) = a_1x_0^n + a_2x_0^{n-1} + \cdots + a_nx_0 + a_{n+1} \tag{3.18}$$

formülünün kodlanması suretiyle hesaplanmaz. Çünkü bu şekliyle n^2 ile orantılı sayıda çarpma işlemi gerçekleştirilmesi gerekmektedir. Örneğin

$$\begin{aligned} P_3(x) &= a_1x^3 + a_2x^2 + a_3x + a_4 \\ &= a_1 \times x \times x \times x + a_2 \times x \times x + a_3 \times x + a_4 \end{aligned}$$

polinomu için $P_3(x_0)$ değerinin hesaplanması 6 adet çarpma işlemi ve 3 adet toplama işlemi gerektirir. Oysa aynı işlem

$$P_3(x) = ((a_1x + a_2)x + a_3)x + a_4$$

örneğinde olduğu üzere iç içe çarpım formatında yazılmak suretiyle 3 adet çarpma ve 3 adet toplama işlemi ile gerçekleştirilebilir. Böylece hem hesaplama işlem zamanında tasarruf sağlanmış olur ve hem de gereksiz aritmetik işlem gerçekleştirmek suretiyle oluşacak yuvarlama hataları engellenmiş olur. Bu durumda

$$b_1 = a_1$$

olarak tanımlanmak üzere

$$\begin{aligned} b_2 &= b_1x_0 + a_2 = a_1x_0 + a_2(\text{en içteki toplam}) \\ b_3 &= b_2x_0 + a_3 = (a_1x_0 + a_2)x_0 + a_3(\text{en içten ikinci toplam}) \\ b_4 &= b_3x_0 + a_4 = ((a_1x_0 + a_2)x_0 + a_3)x_0 + a_4(\text{istenen toplam}) \end{aligned}$$

elde edilmiş olur. Bu işlemi sistematik olarak (3.18) polinomuna genelleştirecek olursak,

$$b_1 = a_1$$

olmak üzere

$$b_k = a_k + x_0b_{k-1}, k = 2, 3, \ldots, n$$

ile tanımlanan $\{b_k\}, k = 1, 2, 3, \ldots, n$ dizisi için yukarıdaki örneğimize paralel olarak

$$b_n = P_n(x_0)$$

olarak elde ederiz. Horner yöntemine ait Algoritma 3.1 aşağıda verilmektedir:

Algoritma 3.1 Horner yöntemi

1. Girdi a, x_0, burada a polinom katsayılarını içeren vektördür.
2. $n \leftarrow a$ nin eleman sayısı
3. $b_1 = a_1$
4. $k = 2, 3, \cdots, n$ için
$$b_k = a_k + x_0 * b_{k-1};$$
5. Çıktı b_n

Düşük dereceli polinomların herhangi bir noktadaki değeri aşağıda belirtilen Tablo yardımıyla daha pratik olarak gerçekleştirilebilir: Örneğin yukarıda tanımlanan $P_3(x)$ polinomunun x_0 noktasındaki değerini hesaplayalım. Yukarıda belirtilen işlemler *Horner*[2] tabosu adı verilen tablo üzerinden kolayca gerçekleştirilebilir.

x_0	a_1	a_2	a_3	a_4
		$x_0 b_1$	$x_0 b_2$	$x_0 b_3$
	$b_1 = a_1$	$b_2 = a_2 + x_0 b_1$	$b_3 = a_3 + x_0 b_2$	$b_4 = a_4 + x_0 b_3$

ÖRNEK 3.5. *$p(x) = x^3 - 2x^2 + x - 4$ polinomunun $x_0 = 1$ noktasındaki değerini Horner yöntemi yardımıyla belirleyiniz.*

Çözüm.

Horner tablosu

1	1	-2	1	-4
		1×1	-1×1	0×1
	1	-1	0	-4

olup, $P(1) = -4$ olarak elde edilir.

Diğer bir bakış açısı ile yukarıda tanımlanan $\{b_k\}, k = 1, 2, 3, \ldots, n$ dizisi için

[2] William George Horner (1786 – 1837, ingiliz matematikçi)

$$\begin{aligned} P_n(x) &= a_1x^n + a_2x^{(n-1)} + \cdots + a_nx + a_{n+1} \\ &= (b_1x^{(n-1)} + \cdots + b_{n-1}x + b_n)(x - x_0) + b_{n+1} \end{aligned}$$

özdeşliği sağlanır ve dolayısıyla $b_{n+1} = P_n(x_0)$ olduğu açıktır.

Her bir b_k nın hesaplanması bir adet çarpma ve bir adet de toplama işlemi gerektirdiğinden Horner yöntemi adı verilen bu yöntemle $n-$inci dereceden bir polinomun herhangi bir noktadaki değerinin hesaplanması n adet çarpma ve n adet toplama işlemi gerektirir. Böylece (3.18) biçiminde $P_n(x_0)$ ın hesaplanması durumunda gereken $O(n^2)$ mertebesindeki işlem, *Horner* yöntemi yardımıyla alternatif olarak n adet çarpma ve n adet toplama işlemi ile gerçekleştirilmiş olmaktadır.

Yüksek dereceli polinomların herhangi bir noktadaki değeri aşağıda verilen ve *horner.m* isimli bir dosyaya kaydedilen Program 3.1 yardımıyla gerçekleştirilebilir.

```
%-----------------------------------------------------------
function sonuc=horner(a,x0);
%x0 skaler
n=length(a);
b=zeros(n,1);
b(1)=a(1);
for k=2:n
  b(k)=a(k)+x0*b(k-1);
end
sonuc=b(n);
%-----------------------------------------------------------
```

Program 3.1: Horner yöntemi(skaler versiyon)

Örneğin $P_2(x) = x^2 - 2x + 3$ polinomunun x_0 noktasındaki değerini hesaplamak için komut ortamından girilen polinom katsayıları ve x_0 noktası için

```
>> a=[1 -2 3]; x0=1;
>> horner(a,x0)
ans =
2
```

elde ederiz.

Tek bir nokta yerine birden fazla noktada verilen bir polinomun değerinin aynı anda hesaplanması istenirse, bu taktirde Program 3.1 ile verilen klasik *Horner* yöntemi geliştirilerek birden fazla noktada polinom değerini hesaplayan algoritma geliştirilmelidir. Birden fazla noktada verilen polinomun değerini hesaplayan algoritma 3.2 aşağıda verilmektedir.

Algoritma 3.2 Horner yöntemi algoritması(vektörel versiyon)

1. Girdi a, x, burada a polinom katsayılarını içeren vektör ve x ise hesaplama noktalarını içeren vektördür
2. $n : a$ nin eleman sayısı; $m \leftarrow x$ in eleman sayısı;
3. $b : n \times m$ boyutlu sıfır matrisi
4. $b_{1,:} = [a(1)a(1)...a(1)$ m bileşenli vektör
5. $k = 2, 3, \cdots, n$ için
$$b_{k,:} = a_k + x_0. * b_{k-1,:};$$
6. Çıktı $b_{n,:}$

Algoritma 3.2 ya ait Program 3.2 aşağıda verilmektedir.

Örneğin $P_2(x) = x^2 - 2x + 3$ polinomunun $x_0 = [1\ 2\ -1]$ vektöründeki değerlerini Program 3.2 ile verilen *vektörel Horner* yöntemi yardımıyla kolayca hesaplayabiliriz:

```
>> a=[1 -2 3];
>> x0=[1 2 -1;]
>> hornerler(a,x0)
ans =
2 3 6
```

```
%-------------------------------------------------------
function sonuc=hornerler(a,x0);
%x0 vektör
%P(x)=a(1)x^n+a(2)x^(n-1)+...+a(n)x+a(n+1)
%sonuc=P(x0)

n=length(a);m=length(x0);
b=zeros(n,m);
b(1,:)=a(1)*ones(1,m);
for k=2:n
  b(k,:)=a(k)+x0.*b(k-1,:);
end
sonuc=b(n,:);
%-------------------------------------------------------
```

Program 3.2: Horner yöntemi(vektörel versiyon)

3.8 İki değişkenli fonksiyonların *Taylor* açılımları

Tek değişkenli fonksiyonlardakine benzer olarak, iki veya daha çok değişkenli fonksiyonların bir nokta komşuluğundaki *Taylor* açılımları elde edilebilir. Örneğin iki değişkenli ve (a,b) noktası komşuluğunda *Taylor* açılımı mevcut olan bir $f(x,y)$ fonksiyonunun bu nokta komşuluğundaki *Taylor* açılımı

$$\begin{aligned} f(x,y) &= f(a,b)+(x-a)\frac{\partial f}{\partial x}|_{(a,b)}+(y-b)\frac{\partial f}{\partial y}|_{(a,b)} \\ &\quad +\frac{1}{2!}\left[(x-a)^2\frac{\partial^2 f}{\partial x^2}|_{(a,b)}+2(x-a)(y-b)\frac{\partial^2 f}{\partial x\partial y}|_{(a,b)}+(y-b)^2\frac{\partial^2 f}{\partial y^2}|_{(a,b)}\right] \\ &\quad +... \end{aligned}$$

olarak elde edilir. Burada

$$P(x,y)=f(a,b)+(x-a)\frac{\partial f}{\partial x}|_{(a,b)}+(y-b)\frac{\partial f}{\partial y}|_{(a,b)}$$

Taylor polinomunun geometrik yeri $z=f(x,y)$ yüzeyine (a,b) noktasında çizilen teğet düzlemdir. $h=x-a$, $k=y-b$ olmak üzere

$$\left(h\frac{\partial}{\partial x}+k\frac{\partial}{\partial x}\right)f(a,b) \;:\; = h\frac{\partial f}{\partial x}|_{(a,b)}+k\frac{\partial f}{\partial y}|_{(a,b)}$$
$$\left(h\frac{\partial}{\partial x}+k\frac{\partial}{\partial x}\right)^2 f(a,b) \;:\; = h^2\frac{\partial^2 f}{\partial x^2}|_{(a,b)}+2hk\frac{\partial^2 f}{\partial x\partial y}|_{(a,b)}+k^2\frac{\partial^2 f}{\partial y^2}|_{(a,b)}$$

notasyonu ile f nin (a,b) noktası komşuluğundaki *Taylor* serisi

$$f(x,y)=\sum_{n=0}^{\infty}\frac{1}{n!}\left(h\frac{\partial}{\partial x}+k\frac{\partial}{\partial x}\right)^n f(a,b) \tag{3.19}$$

olarak ifade edilir.

ÖRNEK 3.6. *$f(x,y)=e^{-(x^2+y^2)}$ fonksiyonunun $(0,0)$ noktası komşuluğundaki, ilk dört Taylor yaklaşım polinomu bularak fonksiyonla birlikte aynı eksende grafiklerini çiziniz.*

Çözüm.

$$\begin{aligned}
f(0,0) &= 1\\
f_x(0,0) &= (-2xe^{-(x^2+y^2)})(0,0)=0\\
f_y(0,0) &= (-2ye^{-(x^2+y^2)})(0,0)=0\\
f_{xx}(0,0) &= e^{-(x^2+y^2)}(4x^2-2)(0,0)=-2\\
f_{yy}(0,0) &= e^{-(x^2+y^2)}(4y^2-2)(0,0)=-2\\
f_{xy}(0,0) &= e^{-(x^2+y^2)}(4xy)(0,0)=0
\end{aligned}$$

değerlerini elde ederiz. Benzer biçimde (3.19) deki diğer gerekli terimleri ve $(0,0)$ daki değerlerini hesaplayarak

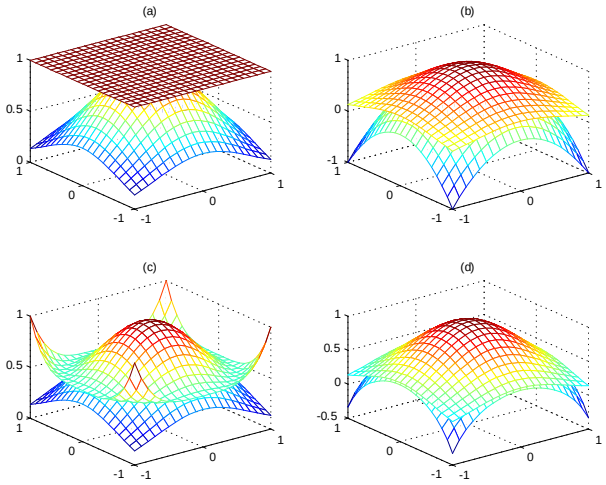
$$f(x,y)=1-(x^2+y^2)+\frac{1}{2!}(x^2+y^2)^2-\frac{1}{3!}(x^2+y^2)^3+...$$

açılımını elde ederiz. (Bu açılımı yukarıda verilen e^{-x^2} açılımı ile karşılaştırınız).

$f(x,y)$ fonksiyonu ve sırasıyla

$$\begin{aligned} P_0(x,y) &= 1, \\ P_2(x,y) &= 1-(x^2+y^2), \\ P_4(x,y) &= 1-(x^2+y^2)+\frac{1}{2!}(x^2+y^2)^2 \\ P_6(x,y) &= 1-(x^2+y^2)+\frac{1}{2!}(x^2+y^2)^2-\frac{1}{3!}(x^2+y^2)^3+... \end{aligned}$$

kısmi toplamlarını(iki değişkenli *Taylor* polinomlarının) grafikleri Şekil 3.3 de sunulmaktadır.



Şekil 3.3: $e^{-(x^2+y^2)}$ fonksiyonu ve artan n değerleri için Taylor polinom yaklaşımları

Artan n değerleri için $P_0(x,y), P_2(x,y), P_4(x,y)$ ve $P_6(x,y)$ polinomlarının grafiklerinin f fonksiyonunun grafiğine yaklaştığı gafiksel olarak görülmektedir.

3.9 *Taylor* açılımı ile aritmetik işlemlerde yuvarlama hatası birikim analizi

x_f değeri x için bir yaklaşım olsun ve $y=f(x)$ fonksiyonu verilsin. x yerine x_f yaklaşımının kullanılması durumunda oluşacak olan $\Delta y = y_f - y$ mutlak hatası ve $\varepsilon_b(y)$ bağıl hatasını tahmin etmek istiyoruz.

Taylor açılımı yardımıyla $\Delta y \cong f'(x)\Delta x$ olarak yazarız. Burada $\cong$ yaklaşımı birinci mertebeden türeve kadar ilgili terimlerin eşitliğini ifade etmektedir. $y \neq 0$ için her iki tarafı y ye bölmek suretiyle

$$\varepsilon_b(y) = \frac{\Delta y}{y} \cong \frac{f'(x)}{f(x)}\Delta x = \frac{f'(x)}{f(x)} x\varepsilon_b(x)$$

elde ederiz. Yukarıdaki son eşitlikte mutlak hata ile bağıl hata arasındaki $\Delta x = x\varepsilon_b(x)$ bağıntısını kullandık.

ÖRNEK 3.7. *x yerine x_f yaklaşımı alınması durumunda, $y = f(x) = x^n$ fonksiyonunda oluşan mutlak ve bağıl hataları x deki mutlak ve bağıl hatalar cinsinden hesaplayınız.*

Çözüm.

Mutlak hata

$$\Delta y \cong f'(x)\Delta x = nx^{n-1}\Delta x$$

olur. Bağıl hata ise

$$\varepsilon_b(y) \cong \frac{f'(x)}{f(x)} x\varepsilon_b(x) = \frac{nx^{n-1}}{x^n} x\varepsilon_b(x) = n\varepsilon_b(x)$$

olarak elde edilir.

ÖRNEK 3.8. *x yerine x_f yaklaşımı alınması durumunda, $y = f(x) = \sqrt[n]{x}$ fonksiyonunda oluşan mutlak ve bağıl hataları x deki mutlak ve bağıl hatalar cinsinden hesaplayınız.*

Çözüm.

Mutlak hata

$$\Delta y \cong f'(x)\Delta x = \frac{1}{n} x^{\frac{1}{n}-1}\Delta x$$

olur. Bağıl hata ise

$$\varepsilon_b(y) \cong \frac{f'(x)}{f(x)} x\varepsilon_b(x) = \frac{\frac{1}{n}x^{(\frac{1}{n}-1)}}{x^{1/n}} x\varepsilon_b(x) = \frac{1}{n}\varepsilon_b(x)$$

olarak elde edilir.

Benzer biçimde iki değişkenli bir $z = f(x, y)$ fonksiyonu için x_f, y_f değerleri x ve y için birer yaklaşım olsunlar. Bu durumda

$$\Delta z \cong \frac{\partial f}{\partial x}\Delta x + \frac{\partial f}{\partial y}\Delta y$$

ve

$$\varepsilon_b(z) \cong \frac{x}{f(x,y)}\frac{\partial f}{\partial x}\varepsilon_b(x) + \frac{y}{f(x,y)}\frac{\partial f}{\partial y}\varepsilon_b(y)$$

elde ederiz. Yüksek mertebeden kısmi türevlerin özdeş olarak sıfıra eşit olduğu durumlarda yukarıda verilen mutlak ve bağıl hata bağıntılarında '$\cong$' yerine '$=$'alınmalıdır.

Buna göre özel olarak toplama, çıkarma, çarpma ve bölme işlemleri için sırasıyla $z = f(x,y)$ fonksiyonunu

$$f(x,y) = x+y, x-y, x\times y, x/y$$

almak suretiyle

toplama/çıkarma	$z = f(x,y) = x \pm y$
	$\Delta z = \Delta x \pm \Delta y$
	$\varepsilon_b(z) = \frac{1}{x\pm y}(x\varepsilon_b(x) \pm y\varepsilon_b(y), x \pm y \neq 0$

elde ederiz. Fark için elde edilen yukarıdaki bağıntıdan, birbirine yakın sayıların farkının alınması sonucu oluşan bağıl hatanın x_f ve y_f yaklaşımlarında oluşan bağıl hatalardan çok daha büyük olduğunu görmekteyiz. Benzer biçimde çarpma ve bölme için aşağıda verilen mutlak ve bağıl hata ifadelerini elde ederiz:

çarpma	$z = f(x,y) = x \times y$
	$\Delta z = y\Delta x + x\Delta y$
	$\varepsilon_b(z) = \varepsilon_b(x) + \varepsilon_b(y)$

bölme	$z = f(x,y) = x/y$
	$\Delta z \cong \frac{1}{y}\Delta x - \frac{x}{y^2}\Delta y$
	$\varepsilon_b(z) \cong \varepsilon_b(x) - \varepsilon_b(y)$

ÖRNEK 3.9. *Karesel bir ofisin bir kenarı, maximum* 0.1m *hata ile* $x = 3m$ *olarak ölçülmüş olsun. Bu değer ile ofis alanının hesaplanmasında oluşacak olan mutlak ve bağıl hata yaklaşık olarak ne kadardır?*

Çözüm.

$y = f(x) = x^2$ alınırsa

$$\Delta y \cong f'(x)\Delta x = 2x\Delta x = 2 \times 3 \times (0.1) = 0.6$$

elde ederiz.

Bağıl hatayı ise yaklaşık olarak

$$\varepsilon_b(y) = \frac{\Delta y}{y} = 0.6/9 \doteq 0.0667$$

olarak elde ederiz.

ÖRNEK 3.10. *Dikdörtgensel bir ofisin taban boyutları sırasıyla makimum* 0.1m *ve* $0.2m$ *hata ile* $x = 3m$ *ve* $y = 5m$ *olarak ölçülmüş olsun. Bu değerlerle ofis alanının hesaplanmasında oluşacal olan mutlak ve bağıl hata yaklaşık olarak ne kadardır?*

Çözüm.

$$z = f(x, y) = x \times y$$

alınırsa

$$\Delta z = y\Delta x + x\Delta y = 5 \times (0.1) + 3 \times (0.2) = 1.1$$

elde ederiz.

Bağıl hatayı ise

$$\varepsilon_b(z) = \frac{\Delta z}{z} = 1.1/15 = 0.0733$$

olarak elde ederiz. Elde edilen bu sonucun x ve y de oluşan bağıl hataların toplamına, yani

$$\varepsilon_b(x) + \varepsilon_b(y) = \frac{\Delta x}{x} + \frac{\Delta y}{y} = 0.1/3 + 0.2/5 = 1.1/15$$

eşit olduğuna dikkat edelim.

ÖRNEK 3.11. *Sürtünmesiz ortamda hareket eden bir cismin kütlesinin maksimum* 0.1 kg *hata ile* $m = 2\ kg$ *ve cisme* t *anında etki eden kuvvetin ise maksimum* $0.2kg \times m/s^2$ *hata ile* $F = 5(kg \times m/s^2)$ *olarak ölçüldüğünü kabul edelim. Bu değerleri kullanmak suretiyle cismin ivmesinin hesaplanmasında oluşacak olan mutlak ve bağıl hata yaklaşık olarak ne kadardır?*

Çözüm.

$\Delta m = 0.1$ ve $\Delta F = 0.2$ mutlak hataları ve $II.$ *Newton* yasası gereği $a = F/m = 2.5(m/s^2)$ ivmesinde oluşan mutlak hata

$$\Delta a \cong \frac{1}{m}\Delta F - \frac{F}{m^2}\Delta m = \frac{1}{2}(0.2) - \frac{5}{2^2}(0.1) = -0.0250$$

dir. Ayrıca

$$\varepsilon_b(m) = \Delta m/m = 0.1/2 = 0.05$$

ve

$$\varepsilon_b(F) = \Delta F/F = 0.2/5 = 0.04$$

olup,

$$\varepsilon_b(a) \cong \varepsilon_b(F) - \varepsilon_b(m) = 0.04 - 0.05 = -0.01$$

olarak elde edilir. Öteyandan

$$\varepsilon_b(a) = \frac{\Delta a}{a} \cong \frac{-0.0250}{2.5} = -0.01$$

olarak aynı sonuç elde edilir.

ÖRNEK 3.12.

$$f(a,b,c) = -b + \sqrt{b^2 - 4ac}$$

fonksiyonunu göz önüne alalım. $a = 1$, $b = 2$, $c = 0.001$ *için, mutlak hata değerlerlerinin en fazla* $\Delta a = 0.1$, $\Delta b = 0.1$, $\Delta c = 0.001$ *olduğunu kabul edelim.* f *nin* (a,b,c) *noktasındaki değerinin hesaplanmasında oluşacak olan mutlak ve bağıl hata yaklaşık olarak ne kadardır?*

Çözüm.

$$\begin{aligned}
\Delta f &\cong \frac{\partial f}{\partial a}\Delta a + \frac{\partial f}{\partial b}\Delta b + \frac{\partial f}{\partial c}\Delta c \\
&= \frac{-2c}{\sqrt{b^2-4ac}}\Delta a + (-1 + \frac{b}{\sqrt{b^2-4ac}})\Delta b + \frac{-2a}{\sqrt{b^2-4ac}}\Delta c \\
&= (-0.0010) \times 0.1 + (5.0038e-004) \times 0.1 + (-1.0005) \times 0.001 \\
&= -0.00105046
\end{aligned}$$

elde ederiz. Öte yandan

$$f(a,b,c) = f(1,2,0.001) = -0.00100025$$

olup,

$$\varepsilon_b(f) \cong \frac{\Delta f}{f} = \frac{0.00105046}{0.00100025} = 1.05019745$$

elde ederiz. Bağıl hatanın mutlak hataya kıyasla daha büyük olduğuna dikkat edelim. Bunun nedeni verilen a, b ve c değerleri için $f(a,b,c)$ nin birbirine yakın iki sayının farkını alan bir işlem olmasıdır. Daha önceden de tahmin ettiğimiz gibi birbirine yakın iki sayının farkının hesaplanmasında bağıl hata büyük olmaktadır.

Alıştırmalar 3.1.

1. *Aşağıda verilen fonksiyonların $a = 0$ noktasında hesaplanan Taylor açılımlarının doğruluğunu kontrol ediniz.*

   - $$\sinh(x) = x + x^3/6 + x^5/120 + x^7/5040 + ...$$
   - $$\cosh(x) = 1 + x^2/2 + x^4/24 + x^6/720 + ...$$
   - $$\tan(x) = x + x^3/3 + 2x^5/15 + 17x^7/315 + ...$$
   - $$\sqrt{1+x} = 1 + x/2 - x^2/8 + x^3/16 - 5x^4/128 + ...$$

2. *Aşağıda verilen fonksiyonların $a = 1$ noktasında hesaplanan Taylor açılımlarının doğruluğunu kontrol ediniz.*

   - $$x^5 = 1 + 5(x-1) + 10(x-1)^2 + 10(x-1)^3 + ...$$
   - $$\begin{aligned} sin(x) &= sin(1) + cos(1)(x-1) - sin(1)(x-1)^2/2 \\ &\quad -cos(1)(x-1)^3/6 + sin(1)(x-1)^4/24 \\ &\quad +cos(1)(x-1)^5/120 + ... \end{aligned}$$

- $$\begin{aligned} sin(5x) &= sin(5) + 5cos(5)(x-1) - 25sin(5)(x-1)^2/2 \\ &\quad -125cos(5)(x-1)^3/6 + 625sin(5)(x-1)^4/24 + ... \end{aligned}$$

- $$\ln(\mathsf{x}) = \mathsf{x} - 1 - (\mathsf{x}-1)^2/2 + (\mathsf{x}-1)^3/3 - (\mathsf{x}-1)^4/4 + (\mathsf{x}-1)^5/5 + ...$$

3. *Terim terime türev veya integral almak suretiyle veya trigonometrik özdeşliklerden faydalanarak aşağıda sol sütunda yer alan fonksiyonların $a = 0$ noktasındaki Taylor açılımlarını, sağ sütunda yer alan fonksiyonların aynı noktadaki açılımları yardımıyla hesaplayınız ve aynı yakınsaklık yarıçaplarına sahip sahip olduklarını gösteriniz.*

   - $sin(x)$, $cos(x)$
   - $tan^2(x)$, $tan(x)$
   - $cos^2(x)$, $\cos(2x)$
   - $1/(1+x)$, $\ln(1+x), \log(1+x)$
   - $cosh(x), \mathsf{e}^x$

4. *Sıfır noktası komşuluğunda bilinen açılımlar yardımıyla aşağıdaki açılımların doğruluğunu kontrol ediniz.*

   - $$\mathsf{x}\sin(\mathsf{x}) = \mathsf{x}^2 - \mathsf{x}^4/6 + \mathsf{x}^6/120 + ...$$

   - $$\ln(1+\mathsf{x}^2) = \mathsf{x}^2 - \mathsf{x}^4/2 + \mathsf{x}^6/3 + ...$$

   - $$\tan(\mathsf{x})/\mathsf{x} = 1 + 1/3\mathsf{x}^2 + 2/15\mathsf{x}^4 + 17/315\mathsf{x}^6 + ...$$

   - $$(1+\mathsf{x})/(1-\mathsf{x}) = 1 + 2\mathsf{x} + 2\mathsf{x}^2 + 2\mathsf{x}^3 + 2\mathsf{x}^4 + 2\mathsf{x}^5 + 2\mathsf{x}^6 + ...$$

5. *Taylor polinom yaklaşımı için verilen hata formülünü kullanarak aşağıda verilen f fonksiyonlarına, $a = 0$ noktasındaki P_n Taylor polinomları ile $[-1, 1]$ aralığında yaklaşıldığını kabul edelim. $||f - P_n||_\infty < \epsilon = 0.1$ eşitsizliği sağlanacak biçimdeki en küçük n tamsayıları sırasıyla ne olmalıdır?*

- $\sin(x)$
- $\cos(x)$
- $\exp(-x)$

6. *Soru 5 i $[-2,2]$ aralığı için tekrarlayınız. Elde ettiğiniz sonucu Soru 5 teki cevabınızla karşılaştırınız. Ne gözlemliyorsunuz?*

7. $$\int_{-1}^{1} \sin(x^2)dx \cong 0.6205$$

integralini hesaplamak istediğimizi düşünelim. P_n polinomu $a=0$ noktası komşuluğunda fonksiyonun $n-$inci dereceden Taylor polinomu olmak üzere $n=0,1,2,...$ için

$$I_n = \int_{-1}^{1} P_n(x)dx$$

integraller dizisini hesaplayalım. I_n dizisinin yakınsadığı noktayı belirleyiniz. Elde ettiğiniz limit verilen integral için iyi bir yaklaşım mıdır?

8. *$f(x)=cos(x)$ fonksiyonu için $a=0$ noktası komşuluğunda elde edilen P_n Taylor polinomlarını göz önüne alalım. $n=0,2,4$ için $[-\pi,\pi]$ aralığında $E_n=||f-P_n||_\infty$ normlarını hesaplayınız. E_n değerleri nasıl değişmektedir?*

9. *$f(t)=e^{-1/t^2}, t\neq 0, f(0)=0$ ile tanımlanan fonksiyonun $t=0$ noktasında bütün basamaktan türevlerinin mevcut ve sıfıra eşit olduğunu sıfır noktasındaki türevin tanımını kullanmak suretiyle gösteriniz. f nin sıfır noktasındaki Taylor seri açılımı, sıfır noktasının komşuluğunda f yi temsil eder mi?*

10. *Integraller için ortalama değer teoremi yardımıyla*

$$R_n(x) = \frac{1}{n!}\int_{x_0}^{x} f^{(n+1)}(t)(t-x_0)^n dt = (x-x_0)^{n+1}/(n+1)!f^{(n+1)}(c_x)$$

sağlanacak biçim de x ile x_0 arasında c_x olduğunu gösteriniz.

11. *$P_n(x)$ polinomu $f(x)$ fonksiyonunun $x = a$ noktasındaki $n-$inci dereceden Taylor polinomu olsun.*

$$P_n^{(k)}(a) = f^{(k)}(a), k = 0, 1, \cdots, n$$

olduğunu gösteriniz.

12. *Aşağıda verilen iki değişkenli fonksiyonların $(0, 0)$ noktasındaki Taylor açılımlarının doğruluğunu kontrol ediniz*

- $$\cos(xy) = 1 - (xy)^2/2 + (xy)^4/24 + ...$$
- $$\cos(x + y) = 1 - (x + y)^2/2 + (x + y)^4/24 + ...$$
- $$\log(x + y + 1) = x + y - (x + y)^2/2 + (x + y)^3/3 - (x + y)^4/4 + ...$$

13. *Kağıt-kalem ve Horner programı yardımıyla aşağıda verilen polinomların belirtilen noktalardaki değerlerini hesaplayınız*

- $$P(x) = x^3+9x^2+27x + 27, x_0= -1$$
- $$P(x) = x^4-3x^3+2x^2-4x + 1, x_0= 1$$
- $$P(x) = x^5-10x^4+40x^3-80x^2+80x - 32, x_0= 3$$
- $$P(x) = x^6-21x^5+175x^4-735x^3+1624x^2-1764x + 720, x_0= 1$$

14. *Vektör tabanlı Horner yöntemi yardımıyla aşağıda verilen polinomların verilen noktalardaki değerlerini eş zamanlı olarak* **vektör cebirsel** *işlemler yardımıyla belirleyiniz.*

- $$P(x) = x^3-6x^2+8x, x_0= [1\ 2\ 4\ 5];$$

- $$\mathrm{P(x)} = \mathrm{x}^4 - 10\mathrm{x}^3 + 35\mathrm{x}^2 - 50\mathrm{x} + 24; \mathrm{x}_0 = [1.2\ 2.5\ 3.4];$$

15. *Horner programı ile aynı formatta çalışan Octave polyval komutu ile soru 12 ve 13 deki polinom değerlerini hesaplayınız.*

16. $g = 9.8, v_0 = 4, x_0 = 2$ *olmak üzere* $y = 1/2gt^2 + v_0 t + x_0$ *değerinin enfazla* $\Delta t = 0.1$ *hatasına sahip* $t = 1$ *ölçüm anındaki değerinin hesaplanmasında oluşacak olan mutlak ve bağıl hata yaklaşık olarak ne kadardır?*

17. $a = 2, b = 4$ *ve* $c = 1$ *değerleri için maksimum mutlak hatalar sırasıyla* $0.1, 0.2$ *ve* 0.3 *olmak üzere olmak üzere*

$$x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \ ve \ x_2 = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$$

değerlerinin hesaplanmasında oluşacak olan mutlak ve bağıl hata yaklaşık olarak ne kadardır?

Projeler 3.1.

1. *Maxima veya ticari Maple, Mathematica ve MATLAB sembolik araç kutusu gibi sembolik cebir programları yardımıyla Taylor polinomları hesaplanabilir. Maxima programını bilgisayarınıza kurarak, aşağıdaki komutlar yardımıyla Taylor polinomlarının hesaplandığını ve grafiklerinin çizdirildiğini gözlemleyiniz.*

   - $taylor(\sin(x), x, 0, 5);$
   - *(%o5)/T/ x-x^3/6+x^5/120+...*
   - */* [wxMaxima: input start] */*
   - $\mathrm{wxplot2d}([x, x - x\hat{}3/6, x - x\hat{}3/6 + x\hat{}5/120, \sin(x)], [x, -4, 4])\$$
   - */* [wxMaxima: input end] */*

2. $$P_2(a) = f(a), P_2'(a) = f'(a), P_2''(a) = f''(a)$$

   özelliğini sağlayan ve x *in kuvvetleri cinsinden yazılan*

$$P_2(x) = c_2 x^2 + c_1 x + c_0$$

polinomunun katsayılarının

$$c0 = (a^2 f''(a) - 2af'(a) + 2f(a))/2, c1 = f'(a) - af''(a), c2 = f''(a)/2$$

olduğunu gösteriniz. $a = 0$ *olması durumunda katsayıların bilinen değerler olduğunu, yani*

$$P_2(x) = 1/2f''(0)x^2 + f'(0)x + f(0)$$

olduğunu gözlemleyiniz.

3. *Proje 2 den biraz farklı olarak,* $[a, b]$ *aralığında* f *fonksiyonuna daha iyi bir yaklaşım olacağı düşünülerek*

$$Q_2(a) = f(a), Q_2'(a) = f'(a), Q_2(b) = f(b)$$

özelliklerini sağlayan

$$Q_2(x) = c_2 x^2 + c_1 x + c_0$$

polinomunun katsayılarının

$$\begin{aligned} c_0 &= -\frac{ab(b-a)f'(a) - a^2 f(b) + (2a-b)bf(a)}{(b-a)^2} \\ c_1 &= \frac{(b^2-a^2)f'(a) + 2a(f(a) - f(b))}{(b-a)^2} \\ c_2 &= -\frac{(b-a)f'(a) - f(b) + f(a)}{(b-a)^2} \end{aligned}$$

olduğunu gösteriniz. Özel olarak $a = 0$ *için*

$$Q_2(x) = -\frac{(b)f'(0) - f(b) + f(0)}{b^2}x^2 + f'(0)x + f(0)$$

olduğunu gözlemleyiniz.

4. $f(x) = sin(x)$ *fonksiyonuna* $[0, 1]$ *aralığıda yaklaşım için kullanılmak üzere Proje 2 deki* $P_2(x)$ *ve Proje 3 deki* $Q_2(x)$ *polinomlarını belirleyerek her bir polinom için*

$$E_1(x) = sin(x) - P_2(x),$$

$$E_2(x) = sin(x) - Q_2(x)$$

hatasının grafiğini 0.1 *aralıklı noktalar için çizdiriniz. Her bir yaklaşım için* $||E_1(x)||_\infty$ *ve* $||E_2(x)||_\infty$ *hatalarını tahmin ediniz. Hangi yaklaşım daha iyidir?*

5. $Q_3(x)$ *polinomu,* $[0, 1]$ *aralığında bir* f *fonksiyonuna yaklaşım için oluşturulan ve*

$$Q_3(0) = f(0), Q_3'(0) = f'(0), Q_3(1) = f(1), Q_3'(1) = f'(1)$$

özelliklerini sağlayan

$$Q_3(x) = c_3x^3 + c_2x^2 + c_1x + c_0$$

poinomunun katsayıları ile f *ve türevinin* $x = 0$ *ve* $x = 1$ *noktasındaki değerleri cinsinden hesaplayınız.* $Q_3(x)$ *polinomunu*

$$P_3^{(k)}(0) = f^{(k)}(0), k = 0, 1, 2, 3$$

özelliklerini sağlayan $a = 0$ *noktası komşuluğundaki* 3*-üncü dereceden Taylor polinomuna alternatif bir polinom veya iki noktalı Taylor polinom açılımı olarak düşünebiliriz. Proje 4 ü,* $P_3(x)$ *ve* $Q_3(x)$ *için tekrarlayınız.*

6. *MATLAB GUI adı verilen arayüzler geliştirmek suretiyle arka planda MATLAB komutlarını çalıştıran etkileşimli programlar hazırlanabilir. Kullanıcı tarafından girilen fonksiyon, açılımın etrafında gerçekleşeceği nokta ve bu noktayı içeren aralık ile istenilen Taylor polinomunun derecesini alarak, Taylor polinomunun ve fonksiyonun grafiğini aynı eksende çizdiren bir GUI hazırlayınız. Ayrıca hazırlayacağınız arayüzde Taylor polinomunun yaklaşım davranışını belirleyen*

$$||f - P_n||_\infty$$

normu da göstermelidir.

Kaynaklar

[1] Atkinson, K. An Introduction to Numerical Analysis, John Wiley & Sons, 1988.

[2] Boas, M. L., Mathematical Methods in the Physical Sciences, John Wiley & Sons, 1983.

[3] Coşkun, E. OCTAVE ile Sayısal Hesaplama ve Kodlama(URL:erhancoskun.com.tr).

[4] Coşkun, E. Maxima ile Sembolik Hesaplama ve Kodlama(URL:erhancoskun.com.tr).

[5] Davis, J. P. ve Rabinowitz, P., Methods of Numerical Integration, Academic Press, 1983.

[6] Kincaid, D., Cheney, W., Numerical Analysis, Brooks/Cole, 1991.

[7] Stoer, J., Bulirsh, R., Introduction to Numerical Analysis, Springer-Verlag, 1976.

[8] S. W., Warren, Zill, D. G., Calculus: Early Transcendentals, Çeviri: Matematik Cilt I, II(Çeviri editörü İsmail Naci Cangül), Nobel Akademik Yayıncılık, 2010.

Bölüm 4

Polinomlarla Interpolasyon ve hata

Bu bölümde bir fonksiyon veya ölçüm sonucu elde edilen nokta kümesinden geçen polinomu belirlemek suretiyle, veri kümesi içerisinde eksik olan değeri tahmin etme işlemi olarak tanımlanan *interpolasyon* kavramını tanıtarak,

- interpolasyon polinomunu adı verilen söz konusu polinomun Cebirsel, Newton ve Lagrange gibi farklı formülasyonlar yardımıyla elde edilişini,
- farklı formülasyonların özelliklerini,
- elektronik ortamda interpolasyon işleminin nasıl gerçekleştirildiğini ve
- interpolasyon işleminde oluşan hata kavramlarını MATLAB/Octave'ın **vektör cebirsel işlem** yetenekleri yardımıyla inceliyoruz.

Konunun **skaler cebirsel** analiz için [1],[3],[4],[5] ve [6] nolu temel referans kaynaklarını öneririz.

4.1 Giriş

Herhangi bir araştırma sonucunde belirli bir amaca yönelik olarak elde edilmiş olan ve genelliği bozmaksızın apsisleri küçükten büyüğe doğru sıralı ve birbirlerinden farklı olan

$$A = \{(t_i, y_i) | i = 0, 1, ..., n\}$$

nokta çiftleri kümesinin verildiğini kabul edelim. A kümesindeki verilerden hareketle, herhangi bir

$$t \in [t_0, t_n], t \neq t_i, i = 0, 1, ..., n$$

noktasındaki bilinmeyen y değerini tahmin etme işlemi sıkça ihtiyaç duyulan bir işlemdir ve bu işleme *interpolasyon işlemi* adı verilmektedir.

Fiziksel uygulamalarda t_i değerleri deney gözlem zamanlarını ve y_i 'ler ise bu zamanlarda elde edilen deneysel gözlem sonuçları olabilir. Bu durumda interpolasyon, ölçüm yapılmamış olan bir t zamanındaki değeri tahmin işlemidir. Benzer biçimde eğer t_i ler nüfus sayımı gerçekleştrilen yıllar ise, sayım yapılmamış olan herhangi $t \in [t_0, t_n]$ yılına ait nüfusu tahmin işlemi bir interpolasyon işlemidir. Gözönüne alınan veri aralığının, yani $[t_0, t_n]$ aralığının dışarısındaki bir noktadaki tahmin işlemi ise *ekstrapolasyon* olarak adlandırılmakta olup, çalışmamızda incelenmemektedir.

t noktasındaki bilinmeyen değeri tahmin etmek için

$$P(t_i) = y_i, i = 0, 1, ..., n \tag{4.1}$$

özelliğini sağlayan ve interpolasyon polinomu adı verilen derecesi en küçük P polinomu bulunarak, bilinmeyen $P(t)$ değeri bu polinom yardımıyla tahmin edilebilir. Bu noktada bir çok soru akla gelmektedir:

- (4.1) özelliğini sağlayan bir polinom var mıdır?
- söz konusu polinom nasıl belirlenir?
- elde edilen ve belirtilen özellikleri sağlayan en düşük dereceli polinom, yani interpolasyonu polinomu tek midir ve derecesi hakkında ne söyleyebiliriz?
- elde edilen interpolasyon polinomu yardımıyla gerçekleştirilen interpolasyon işleminde oluşan hata tahmini olarak ne kadardır?
- sözkonusu hatayı minimize etmek için neler yapılmalıdır?

Yukarıda belirtilen soruların önemli bir kısmı gerek Newton ve gerekse Lagrange'ın çalışmalarıyla açıklığa kavuşturulmuş ve istenilen özellikleri sağlayan interpolasyon polinomunun varlığı, bu polinomu inşa etmek suretiyle ispatlanmıştır. Kullanılan yöntemler ilerleyen bölümlerde inceleyeceğimiz üzere birbirinden farklıdır.

4.2 İnterpolasyon polinomunun farklı formülasyonları

Öncelikle iki adet $(t_0, y_0), (t_1, y_1)$ nokta çifti gözönüne alalım.

- Bu iki nokta çifti için (4.1) özelliğini sağlayan çok sayıda polinom bulabiliriz. Ancak aklımıza gelen en düşük dereceli ilk polinom birinci dereceden olup, grafiği bu iki noktadan geçen doğrudur:

$$P_1(t) = y_0 + \frac{y_1 - y_0}{t_1 - t_0}(t - t_0) \tag{4.2}$$

- (4.2) polinomu, yukarıda yazıldığı biçimiyle Newton formülasyonudur, ancak şüphesiz ki bu polinomu Newton veya Lagrange'ın geliştirdikleri ileri düzey formülasyonlara ihtiyaç duymadan da belirleyebiliriz:

$$P_1(t) = a + bt \tag{4.3}$$

biçiminde bir polinom arayarak, grafiği söz konusu iki noktadan geçecek şekilde, yani (4.1) özelliği sağlanacak biçimdeki a ve b katsayılarını bulabiliriz:

$$\begin{aligned} a + bt_0 &= y_0 \\ a + bt_1 &= y_1 \end{aligned} \tag{4.4}$$

denklem sistemini çözerek

$$a = y_0 - bt_0,\ b = \frac{y_1 - y_0}{t_1 - t_0} \tag{4.5}$$

elde ederiz. (4.4) siteminin çözümünün varlığı ve tekliği için $t_1 \neq t_0$ olması gerektiğine dikkat edelim.

- (4.2) polinomunun alternatif olarak

$$P_1(t) = \frac{t - t_1}{t_0 - t_1}y_0 + \frac{t - t_0}{t_1 - t_0}y_1 \tag{4.6}$$

biçiminde de yazılabileceğine dikkat edelim. O halde aynı polinomu üç farklı yöntemle elde etmiş olduk:

- Birincisi (4.2) ile verilen **Newton formülasyonudur**. (4.3) ile verilen ikinci formülasyon **Cebirsel formülasyon** adı verilmektedir. Üçüncüsü ise (4.6) ile verilen **Lagrange formülasyonudur**.

4.2.1 Cebirsel formülasyon

Cebirsel formülasyonu daha çok sayıda nokta çiftine genelleştirmek oldukça kolaydır. Örneğin üç adet $(t_0, y_0), (t_1, y_1), (t_2, y_2)$ nokta çifti için grafiği bu noktalardan geçen ve ikinci dereceden olan

$$P_2(t) = a + bt + ct^2$$

polinomunu benzer biçimde belirleyebiliriz: (4.1) özelliğinden,

$$\begin{aligned} a + bt_0 + ct_0^2 &= y_0 \\ a + bt_1 + ct_1^2 &= y_1 \\ a + bt_2 + ct_2^2 &= y_2 \end{aligned}$$

veya

$$\begin{bmatrix} 1 & t_0 & t_0^2 \\ 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ y_2 \end{bmatrix} \tag{4.7}$$

lineer denklem sistemini çözmek suretiyle a, b ve c katsayılarını ve dolayısıyla da istenilen özellikleri sağlayan $P_2(t)$ polinomunu elde ederiz.

$(t_i, y_i), i = 0, 1, ..., n$ ile verilen ve $(i \neq j \Rightarrow t_i \neq t_j,)$ özelliklerini sağlayan $(n + 1)$ adet nokta çifti için ise $n - inci$ dereceden

$$P_n(t) = a_0 + a_1 t + ... + a_{n-1} t^{n-1} + a_n t^n \tag{4.8}$$

polinomu, (4.1) özelliğinden hareketle

$$\begin{bmatrix} 1 & t_0 & \cdots & t_0^n \\ 1 & t_1 & \cdots & t_1^n \\ \vdots & \vdots & \vdots & \vdots \\ 1 & t_n & \cdots & t_n^n \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix} \tag{4.9}$$

denklem sistemi çözülerek (4.8) ile tanımlanan interpolasyon polinomunun katsayıları ve dolayısıyla da istenilen özellikleri sağlayan interpolasyon polinomu teorik olarak elde edilebilir.(4.9) sisteminin katsayı matrisi **Vandermonde** matrisidir ve $i \neq j \Rightarrow t_i \neq t_j$ özelliğinden dolayı determinantı sıfırdan farklıdır(Alıştırma 3), yani (4.9) tek bir çözüme sahiptir.

Dolayısıyla cebirsel formülasyon yardımıyla derecesi $\leq n$ olan ve (4.1) özelliğini sağlayan polinomun **var ve tek** olduğunu görüyoruz.

ÖRNEK 4.1. $(0,0),(1,2),(2,5)$ *noktalarından geçen interpolasyon polinomunu cebirsel formülasyon yardımıyla belirleyiniz.*

Çözüm.

Grafiği, verilen üç noktadan geçen interpolasyon polinomunu

$$P_2(t) = a + bt + ct^2$$

ile gösterelim.

$$P_2(t_i) = y_i, i = 0, 1, 2$$

özelliğinden

$$\begin{aligned} a &= 0 \\ a+b+c &= 2 \\ a+2b+4c &= 5 \end{aligned}$$

elde ederiz. Bu sistemi çözerek, $c = 1/2, b = 3/2$ elde ederiz. O halde aradığımız polinom

$$P_2(t) = \frac{t}{2}(t+3)$$

dir.

ÖRNEK 4.2. *Örnek 4.1 de verilen noktalara ilaveten, grafiği* $(4,1)$ *noktasından da geçen interpolasyon polinomunu belirleyiniz.*

Çözüm.

Sözkonusu polinomu

$$P_3(t) = a + bt + ct^2 + dt^3$$

ile gösterelim.

$$P_3(t_i) = y_i, i = 0, 1, 2, 3$$

özelliğinden

$$\begin{aligned} a &= 0 \\ a+b+c+d &= 2 \\ a+2b+4c+8d &= 5 \\ a+4b+16c+64d &= 1 \end{aligned}$$

sistemini veya $a = 0$ olduğundan hareketle

$$\begin{bmatrix} 1 & 1 & 1 \\ 2 & 4 & 8 \\ 4 & 16 & 64 \end{bmatrix} \begin{bmatrix} b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \\ 1 \end{bmatrix}$$

elde ederiz. Gauss eliminasyon yöntemi yardımıyla

$$\begin{bmatrix} 1 & 1 & 1 & | & 2 \\ 2 & 4 & 8 & | & 5 \\ 4 & 16 & 64 & | & 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 1 & 1 & | & 2 \\ 0 & 2 & 6 & | & 1 \\ 0 & 12 & 60 & | & -7 \end{bmatrix}$$

$$\rightarrow \begin{bmatrix} 1 & 1 & 1 & | & 2 \\ 0 & 2 & 6 & | & 1 \\ 0 & 0 & 24 & | & -13 \end{bmatrix}$$

elde ederiz. Elde edilen ekli matrise karşılık gelen üst üçgensel sistemden

$$\begin{aligned} b + c + d &= 2 \\ 2c + 6d &= 1 \\ 24d &= -13 \end{aligned}$$

elde ederiz. Bu sistemden

$$d = -13/24, c = (1 - 6d)/2 = (1 + 13/4) = 17/8$$

ve

$$b = 2 - 17/8 + 13/24 = \frac{5}{12}$$

elde ederiz. O halde

$$P_3(t) = \frac{5}{12}t + \frac{17}{8}t^2 - \frac{13}{24}t^3$$

dir.

Ancak bu yöntem tercih edilen bir yöntem değildir, çünkü katsayı matrisi Vandermonde matrisi olarak adlandırılan (4.9) sisteminin sayısal çözüm problemi **kararlı olmayan** bir problemdir. Diğer bir değimle çözüm aşamasında oluşan yuvarlama hataları kontrolsüz olarak büyüyebilmekte ve elde edilen sayısal çözüm kabul edilemeyecek büyüklükte hata içerebilmektedir. Bu durumda kullanılması gereken alternatif yaklaşımlar Bölüm 5 te incelenmektedir.

Diğer bir alternatif Newton formülasyonudur:

4.2.2 Newton formülasyonu

İki nokta çifti için (4.2) ile verilen interpolasyon polinomu Newton formunun, $(t_0, y_0), (t_1, y_1), (t_2, y_2)$ nokta çiftlerine de benzer biçimde genelleştirilebilir: Newton bu nokta çiftleri için ise

$$P_2(t) = a + b(t - t_0) + c(t - t_0)(t - t_1) \tag{4.10}$$

formunda bir polinom önermektedir. (4.1) özelliğinin sağlanabilmesi için (4.2) de olduğu üzere

$$a = y_0, b = \frac{y_1 - y_0}{t_1 - t_0}$$

olması gerektiği açıktır.

$$y_2 = P_2(t_2)$$

özelliğinin sağlanması gerektiğinden hareketle c değerini

$$\begin{aligned} c &= -\frac{a + b(t_2 - t_0)}{(t_2 - t_0)(t_2 - t_1)} \\ &= -\frac{y_0 + \frac{y_1 - y_0}{t_1 - t_0}(t_2 - t_0)}{(t_2 - t_0)(t_2 - t_1)} \\ &= \frac{\frac{y_2 - y_1}{t_2 - t_1} - \frac{y_1 - y_0}{t_1 - t_0}}{t_2 - t_0} \end{aligned}$$

olarak elde edebiliriz:

Uygun bir f fonksiyonu için y_i değerlerinin $y = f(t)$ bağıntısıyla tanımlı fonksiyonun t_i noktasındaki değerlerini Newton formülasyonunda tercih edilen notasyon olarak,

$$f_i := f(t_i), i = 0, 1, ..., n$$

ile gösterelim. (4.10) deki katsayıları daha iyi temsil edebilmek için bölünmüş fark adı verilen aşağıdaki notasyonu tanımlayalım:

$$\begin{aligned} f[t_i, t_{i+1}] &: = \frac{f_{i+1} - f_i}{t_{i+1} - t_i} \\ f[t_{i-1}, t_i, t_{i+1}] &: = \frac{f[t_i, t_{i+1}] - f[t_{i-1}, t_i]}{t_{i+1} - t_{i-1}}, \\ &\vdots \\ f[t_1, t_2, \cdots, t_i] &: = \frac{f[t_2, \cdots, t_i] - f[t_1, t_2, \cdots, t_{i-1}]}{t_i - t_1} \end{aligned}$$

Bu durumda (4.10) da

$$\begin{aligned} a &= f[t_0] := f(t_0) = f_0, \\ b &= f[t_0, t_1] := \frac{f_1 - f_0}{t_1 - t_0}, \\ c &= f[t_0, t_1, t_2] := \frac{f[t_1, t_2] - f[t_0, t_1]}{t_2 - t_0} \end{aligned}$$

olarak ifade edilebilir. Yani (4.10) polinomu

$$\begin{aligned} P_2(t) &= f[t_0] + f[t_0, t_1](t - t_0) + f[t_0, t_1, t_2](t - t_0)(t - t_1) \\ &= P_1(t) + f[t_0, t_1, t_2](t - t_0)(t - t_1) \end{aligned} \tag{4.11}$$

olarak ifade edilir.

En genel halde $(t_i, f_i), i = 0, 1, ..., n$ nokta çiftleri için ise,

$$\begin{aligned} N_0(t) &: = 1, \\ N_1(t) &: = t - t_0, \\ N_2(t) &: = (t - t_0)(t - t_1), \\ &\vdots \\ N_n(t) &: = (t - t_0)(t - t_1) \dots (t - t_{n-1}) \end{aligned}$$

polinomları olmak üzere,

$$\begin{aligned} P_n(t) &= f[t_0] \\ &\quad + f[t_0, t_1](t - t_0) + \cdots \\ &\quad + f[t_0, t_1, \dots, t_n](t - t_0) \dots (t - t_{n-1}) \\ &= P_{n-1}(t) + f[t_0, t_1, \dots, t_n](t - t_0) \dots (t - t_{n-1}) \\ &= \sum_{i=0}^{n} f[t_0, ..., t_i] N_i(t) \end{aligned} \tag{4.12}$$

elde edilir. Az sayıda nokta çiftleri için (4.12) polinomunun katsayıları ise Newton bölünmüş fark tablosu adı verilen Tablo 4.1 yardımıyla daha kolay bir biçimde elde edilir.

Tablo 4.1 deki köşegen üzerindeki elemanların (4.12) interpolasyon polinomunundaki sabit çarpanlar olduğuna dikkat edelim.

t_i	$f[t_i] = f_i$	$f[.,.]$	$f[.,.,.]$	$\cdots$	
t_0	f_0				
t_1	f_1	$f[t_0,t_1]$			
t_2	f_2	$f[t_1,t_2]$	$f[t_0,t_1,t_2]$		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	
t_n	f_n	$f[t_{n-1},t_n]$	$f[t_{n-2},t_{n-1},t_n]$	$\cdots$	$f[t_0,t_1,\ldots,t_n]$

Tablo 4.1: Newton bölünmüş fark tablosu

ÖRNEK 4.3. $(0,0),(1,2),(2,5)$ *noktalarından geçen interpolasyon polinomunu belirleyerek,* $t = 3/2$ *noktasındaki bilinmeyen değeri tahmin ediniz.*

Çözüm.

Öncelikle verilerimize ait Newton bölünmüş fark tablosunu oluşturalım:

t_i	f_i	$f[.,.]$	$f[.,.,.]$
0	0		
1	2	$\frac{2-0}{\mathbf{1-0}} = 2$	
2	5	$\frac{5-2}{\mathbf{2-1}} = 3$	$\frac{3-2}{\mathbf{2-0}} = \frac{1}{2}$

Not: Newton bölünmüş fark tablosu oluşturulurken,

- üçüncü sütunda yer alan ve $f[.,.]$ ile gösterilen ikili farkların paydasında, $1-0, 2-1$ gibi ardışık apsislerin farkı,
- $f[.,.,.]$ ile gösterilen ve dördüncü sütundaki farkların paydasında, $2-0$ gibi tek eleman atlayarak oluşturulan apsisler farkının yer aldığına dikkat edelim.

O halde aradığımız interpolasyon polinomunu

$$\begin{aligned} P_2(t) &= f[t_0] + f[t_0,t_1](t-t_0) + f[t_0,t_1,t_2](t-t_0)(t-t_1) \\ &= 0 + 2(t-0) + \frac{1}{2}(t-0)(t-1) \\ &= 2t + \frac{1}{2}(t^2 - t) \\ &= \frac{t}{2}(t+3) \end{aligned}$$

olarak elde ederiz. Polinom grafiğinin gerçekten de verilen noktalardan geçtiğine dikkat edelim. $t = 1.5$ noktasındaki değeri ise

$$P(3/2) = 3/4 \times (3/2 + 3) = \frac{27}{8}$$

olarak tahmin ederiz.

ÖRNEK 4.4. *Yukarıdaki örnekte verile noktalara ilaveten* $(4, 1)$ *noktasından da geçen interpolasyon polinomunu belirleyiniz ve* $t = 3$ *noktasındaki bilinmeyen değeri tahmin ediniz.*

Çözüm.

t_i	f_i	$f[.,.]$	$f[.,.,.]$	$f[.,.,.,.]$
0	0			
1	2	$\frac{2-0}{\mathbf{1-0}} = 2$		
2	5	$\frac{5-2}{\mathbf{2-1}} = 3$	$\frac{3-2}{\mathbf{2-0}} = \frac{1}{2}$	
4	1	$\frac{1-5}{\mathbf{4-2}} = -2$	$\frac{-2-3}{\mathbf{4-1}} = -\frac{5}{3}$	$\frac{-\frac{5}{3}-\frac{1}{2}}{\mathbf{4-0}} = -\frac{13}{24}$

- $f[.,.,.,.]$ ile gösterilen ve beşinci sütundaki farkların paydasında, $4 - 0$ gibi iki eleman atlayarak oluşturulan apsisler farkının yer aldığına dikkat edelim.

- Verilen $(t_i, f_i), i = 1, 2, ..., n$ çiftleri için Newton bölünmüş fark tablosunu *vektör cebirsel olarak* hesaplayarak D matrisi ile geri gönderen Program 4.1 aşağıda verilmektedir.

  Aradığımız polinom Örnek 4.3 için elde ettiğimiz polinom yardımıyla bulunabilir:

$$\begin{aligned} P_3(t) &= f[t_0] + f[t_0, t_1](t - t_0) + f[t_0, t_1, t_2](t - t_0)(t - t_1) \\ &\quad + f[t_0, t_1, t_2, t_3](t - t_0)(t - t_1)(t - t_2) \\ &= P_2(t) + f[t_0, t_1, t_2, t_3](t - t_0)(t - t_1)(t - t_2) \\ &= \frac{t}{2}(t + 3) - \frac{13}{24}t(t - 1)(t - 2) \\ &= -\frac{13}{24}t^3 + \frac{17}{8}t^2 + \frac{5}{12}t \end{aligned}$$

  Elde edilen polinomun grafiği Şekil 4.1 de sunulmaktadır. Polinom grafiğinin verilen nokta çiftlerinden geçtiğine dikkat edelim.

```
% Newton Bölünmüs Fark Tablosu
% ------------------------------------------------------

function D=bolfark(t,y)
  n=length(t);
  D=zeros(n,n);
  D(:,1)=y;
  for j=2:n
      iv=j:n;
      D(iv,j)=(D(iv,j-1)-D(iv-1,j-1))./(t(iv)-t(iv-j+1))';
                          % Tablonun j-inci sütunu
  end

% ------------------------------------------------------
```

Program 4.1: Newton interpolasyon formülasyonu için bölünmüs fark tablosu

Ayrıca $t = 3$ noktasındaki bilinmeyen değeri

$$P(3) = -\frac{13}{24} \times 3^3 + \frac{17}{8} \times 3^2 + \frac{5}{12} \times 3 = 5.75$$

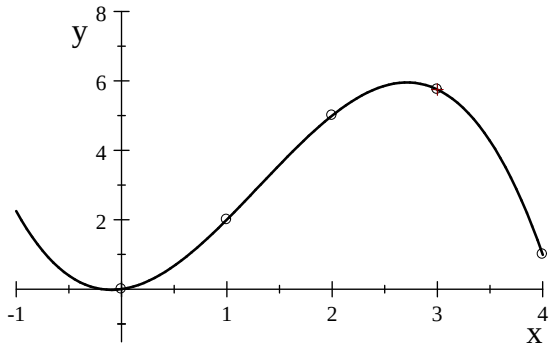
olarak tahmin ederiz.

(t, y) vektör çiftini kullanıcıdan alarak istenilen t_0 noktasındaki değerini yukarıda verilen Newton formülasyonu yardımıyla hesaplayan Algoritma(4.1) aşağıda verilmektedir:

Newton interpolasyon algoritmasına ait Program 4.2 aşağıda verilmektedir.

ÖRNEK 4.5. *t=[0 1 2 4] ve y=[0 2 5 1] için*

>> newinterp(t,y,3)
ans =
5.7500
elde ederiz.
Ayrıca çok sayıdaki interpolasyon noktası için de programımız çalışır. Örneğin
>> tt=[1,2,3]
için
>> newinterp(x,y,tt)



Şekil 4.1: $P_3(t)$ polinomunun grafiği(çizgi) ve nokta çiftleri(o)

ans =
2.0000 5.0000 5.7500
elde ederiz.

4.2.3 Lagrange formülasyonu

İki nokta çifti için (4.6) ile verilen Lagrange formülasyonu, (t_0, y_0),(t_1, y_1),(t_2, y_2) nokta çiftlerine de benzer biçimde genelleştirilebilir. Ancak öncelikle baş harflerini Lagrange isminden alan

$$L_0(t) := \frac{t - t_1}{t_0 - t_1}, L_1(t) := \frac{t - t_0}{t_1 - t_0} \tag{4.13}$$

polinomlarını tanımlayarak (4.6) nin

$$P_1(t) = y_0 L_0(t) + y_1 L_1(t)$$

formunda yazılabileceğine dikkat edelim. Ayrıca $L_0(t)$ ve $L_1(t)$ nin

$$\begin{aligned} L_0(t_0) &= 1, L_0(t_1) = 0; \\ L_1(t_0) &= 0, L_1(t_1) = 0; \end{aligned} \tag{4.14}$$

özelliklerini sağlayan polinomlar olduklarına ve dolayısıyla da

$$P_1(t_0) = y_0,\ P_1(t_1) = y_1$$

Algoritma 4.1 Newton interpolasyon algoritması

1. Girdiler: (t, y) vektör çifti ve tt interpolasyon noktası veya vektörü
2. D bölünmüş fark alt üçgensel matrisini hesapla
   (a) D nin birinci sütununa y vektörünü ata,
   (b) her bir $j = 2, ..., n$ ye karşılık gelen $i = j, j + 1, ...n$ için $D(i, j)$ değerini hesapla,
3. $N(tt) = \{N_1(tt), N_2(tt), \ldots, N_n(tt)\}$ vektör(tt nin nokta olması hali) veya matrisini(tt nin vektör olması hali) hesapla
4. D nin köşegen elemanlarından oluşan vektör ile $N(tt)$ nin iç çarpımı olan $P_n(tt)$ değerini hesapla ve geri gönder.

interpolasyon polinom özelliklerinin sağlandığına dikkat edelim.

$(t_0, y_0), (t_1, y_1), (t_2, y_2)$ nokta çiftleri için aradığımız interpolasyon polinomunun

$$P_2(t) = y_0 L_0(t) + y_1 L_1(t) + y_2 L_2(t) = \sum_{i=0}^{2} y_i L_i(t)$$

formunda olmasını bekleriz. (4.14) e benzer olarak eğer

$$\begin{aligned} L_0(t_0) &= 1, L_0(t_1) = L_0(t_2) = 0; \\ L_1(t_0) &= 0, L_1(t_1) = 1, L_1(t_2) = 0; \\ L_2(t_0) &= L_2(t_1) = 0, L_2(t_2) = 1; \end{aligned} \tag{4.15}$$

özellikleri sağlanırsa bu taktirde

$$P_2(t_i) = y_i, i = 0, 1, 2$$

interpolasyon polinomu özellikleri de sağlanmış olur. Ancak genelde grafiği üç noktadan geçen polinom ikinci dereceden olması gerektiği için bu defa (4.13) dekilerin aksine $L_i(t)$ lerin ikinci dereceden polinomlar olmasını bekleriz. Örneğin $L_0(t)$ nin (4.15) de belirtilen

$$L_0(t_1) = L_0(t_2) = 0$$

```
%  Newton interpolasyon formülasyonu ile aradeğe tahmini
%  t(i),y(i),i=1,2,...,n için Newton interpolasyon
%  polinomunun tt deki değerini veya değerlerini hesaplar
%  ec, Temmuz, 2014
%-------------------------------------------------------

function yy=newinterp(t,y,tt)
n=length(t);m=length(tt);
D=bolfark(t,y);

N=zeros(n,m);
N(1,:)=1;n1=n-1;
for i=1:n1
  N(i+1,:)=N(i,:).*(tt-t(i));
                  %Newton tabanının tt deki değeri(değerleri)
end
for i=1:n
  fark(i)=D(i,i);%bölünmüs farklar köşegen elemanları
end
for i=1:m
 yy(i)=N(:,i)'*fark';
      % interpolasyon polinomunun tt(i) deki değeri
end
%-------------------------------------------------------
```

Program 4.2: Newton interpolasyon formülasyonu ile aradeğer tahmini

özelliğini sağlaması için

$$L_0(t) = c \times (t - t_1)(t - t_2)$$

formunda olması gerekir. Ayrıca $L_0(t_0) = 1$ olması için de

$$L_0(t_0) = c \times (t_0 - t_1)(t_0 - t_2) = 1 \Rightarrow c = \frac{1}{(t_0 - t_1)(t_0 - t_2)}$$

olmalı, yani

$$L_0(t) = \frac{(t - t_1)(t - t_2)}{(t_0 - t_1)(t_0 - t_2)} = \prod_{j=1}^{2} \frac{(t - t_j)}{(t_0 - t_j)}$$

olmalıdır. Benzer biçimde,

$$L_1(t) = \frac{(t-t_0)(t-t_2)}{(t_1-t_0)(t_1-t_2)} = \prod_{\substack{j=0 \\ j\neq 1}}^{2} \frac{(t-t_j)}{(t_0-t_j)}$$

ve

$$L_2(t) = \frac{(t-t_0)(t-t_1)}{(t_2-t_0)(t_2-t_1)} = \prod_{j=0}^{1} \frac{(t-t_j)}{(t_2-t_j)}$$

elde ederiz. Her üç polinomu

$$L_i(t) = \prod_{\substack{j=0 \\ j\neq i}}^{2} \frac{(t-t_j)}{(t_i-t_j)}, i = 0, 1, 2$$

formunda ifade edebiliriz.

$(t_0, y_0), (t_1, y_1), \ldots, (t_n, y_n)$ nokta çiftleri için ise yukarıdaki formülasyonlarda 2 yerine n almak yeterlidir. Yani

$$P_n(t) = \sum_{i=0}^{n} y_i L_i(t), \tag{4.16}$$

$$L_i(t) = \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{(t-t_j)}{(t_i-t_j)}, i = 0, 1, ..., n \tag{4.17}$$

olarak ifade edebiliriz.

ÖRNEK 4.6. *Örnek (4.3) ü Lagrange formülasyonu ile tekrar ediniz. Yani, $(0,0), (1,2), (2,5)$ noktalarından geçen interpolasyon polinomunu Lagrange formülasyonu yardımıyla elde ediniz.*

Çözüm.

$n = 2$ olup, $t_0 = 0, t_1 = 1, t_2 = 2$ için

$$\begin{aligned} L_0(t) &= \frac{(t-t_1)(t-t_2)}{(t_0-t_1)(t_0-t_2)} \\ &= \frac{(t-1)(t-2)}{-1\times(-2)} \\ &= \frac{1}{2}t^2 - \frac{3}{2}t + 1 \\ L_1(t) &= \frac{(t-t_0)(t-t_2)}{(t_1-t_0)(t_1-t_2)} \\ &= \frac{t(t-2)}{1\times(-1)} \\ &= -t^2 + 2t \\ L_2(t) &= \frac{(t-t_0)(t-t_1)}{(t_2-t_0)(t_2-t_1)} \\ &= \frac{t(t-1)}{2\times 1} \\ &= \frac{1}{2}t^2 - \frac{1}{2}t \end{aligned}$$

polinomları yardımıyla

$$\begin{aligned} P_2(t) &= y_0L_0(t) + y_1L_1(t) + y_2L_2(t) \\ &= 0\times L_0(t) + 2\times(-t^2+2t) + 5\times(\frac{1}{2}t^2 - \frac{1}{2}t) \\ &= \frac{t}{2}(t+3) \end{aligned}$$

elde ederiz. Bu polinom daha önce Newton formülasyonu yardımıyla elde ettiğimiz polinomdur.

ÖRNEK 4.7. *Örnek (4.6) ü Lagrange formülasyonu için tekrar ediniz. Yani,* $(0,0),(1,2),(2,5),(4,1)$ *noktalarından geçen interpolasyon polinomunu Lagrange formülasyonu yardımıyla elde ediniz.*

Çözüm.

$n = 3$ olup, $t_0 = 0, t_1 = 1, t_2 = 2, t_3 = 4$ için

$$\begin{aligned} L_0(t) &= \frac{(t-t_1)(t-t_2)(t-t_3)}{(t_0-t_1)(t_0-t_2)(t_0-t_3)} \\ &= \frac{(t-1)(t-2)(t-4)}{(0-1)(0-2)(0-4)} \\ &= -\frac{1}{8}t^3 + \frac{7}{8}t^2 - \frac{7}{4}t + 1 \end{aligned}$$

$$\begin{aligned} L_1(t) &= \frac{(t-t_0)(t-t_2)(t-t_3)}{(t_1-t_0)(t_1-t_2)(t_1-t_3)} \\ &= \frac{(t-0)(t-2)(t-4)}{(1-0)(1-2)(1-4)} \\ &= \frac{1}{3}t^3 - 2t^2 + \frac{8}{3}t \end{aligned}$$

$$\begin{aligned} L_2(t) &= \frac{(t-0)(t-1)(t-4)}{(t_2-t_0)(t_2-t_1)(t_2-t_3)} \\ &= \frac{(t-0)(t-1)(t-4)}{(2-0)(2-1)(2-4)} \\ &= -\frac{1}{4}t^3 + \frac{5}{4}t^2 - t \end{aligned}$$

$$\begin{aligned} L_3(t) &= \frac{(t-t_0)(t-t_1)(t-t_2)}{(t_3-t_0)(t_3-t_1)(t_3-t_2)} \\ &= \frac{(t-0)(t-1)(t-2)}{(4-0)(4-1)(4-2)} \\ &= \frac{1}{24}t^3 - \frac{1}{8}t^2 + \frac{1}{12}t \end{aligned}$$

elde ederiz.

O halde

$$\begin{aligned} P_3(t) &= y_0L_0(t) + y_1L_1(t) + y_2L_2(t) + y_3L_3(t) \\ &= 0 \times L_0(t) + 2 \times L_1(t) + 5 \times L_2(t) + 1 \times L_3(t) \\ &= 2 \times (\frac{1}{3}t^3 - 2t^2 + \frac{8}{3}t) \\ &\quad +5 \times (-\frac{1}{4}t^3 + \frac{5}{4}t^2 - t) \\ &\quad +1 \times (\frac{1}{24}t^3 - \frac{1}{8}t^2 + \frac{1}{12}t) \\ &= -\frac{13}{24}t^3 + \frac{17}{8}t^2 + \frac{5}{12}t \end{aligned}$$

dir. Lagrange interpolasyon algoritması ile aradeğer tahmini Algoritma 4.2 ile verilmektedir.

Algoritma 4.2 Lagrange interpolasyon algoritması

1. Girdiler: (t, y) vektör çifti ve tt interpolasyon noktası veya vektörü
2. $n : t$ nin eleman sayısı ve m ise tt nin eleman sayısı olmak üzere $L_i(tt)$ yi hesapla
   (a) Her bir $i = 1, 2, ..., n$ ve $j = 1, 2, ..., m$ ve $j \neq i$ için
   (b) $L_i(tt)$ değerini 4.17 ile hesapla,
3. $L(tt) = \{L_1(tt), L_2(tt), \ldots, L_n(tt)\}$ vektör(tt nin nokta olması hali) veya matrisini(tt nin vektör olması hali) hesapla
4. y vektörü ile $L(tt)$ nin iç çarpımı olan $P_n(tt)$ değerini hesapla ve geri gönder.

Algoritma 4.2 ya ait Program 4.3 aşağıda verilmektedir.

ÖRNEK 4.8. *>> t=[0 1 2 4] ve >> y=[0 2 5 1] için*

```
>> laginterp(t,y,3)
ans =
5.7500
```

```
function yy=laginterp(t,y,tt)
 % Lagrange interpolasyon formülasyonu ile aradeğer tahmini
 % t(i),y(i),i=1,2,...,n için Lagrange interpolasyon
 % polinomunun tt deki degerini veya değerlerini  hesaplar
 % ec, Temmuz, 2014
 %------------------------------------------------------------

n=length(t);m=length(tt);
L=zeros(n,m);
for i=1:n
  carp=ones(1,m);
  for j=1:n
   if j~=i
      carp=carp.*(tt-t(j))./(t(i)-t(j));
  end
   L(i,:)=carp;
  end
end
for j=1:m
 yy(j)=L(:,j)'*y'; % interpolasyon polinomunun
end                %  tt(j) deki değeri
%------------------------------------------------------------
```

Program 4.3: Lagrange interpolasyon formülasyonu ile aradeğer tahmini

```
>> laginterp(t,y,[1 3])
ans =
2.0000 5.7500
>> laginterp(t,y,[1 2 3])
ans =
2.0000 5.0000 5.7500
```

elde ederiz.

Özetle

- Örnek 4.1 ve 4.2 de sırasıyla üç ve dört nokta çiftine ait interpolasyon polinomunu cebirsel formülasyon yardımıyla belirledik.
- Örnek 4.3 ve 4.4 de aynı nokta çiftlerine ait interpolasyon polinomunu Newton formülasyonu yardımıyla belirledik.

- Örnek 4.6 ve 4.7 te ise bu defa aynı nokta çiftleri için Lagrange formülasyonu yardımıyla ilgili interpolasyon polinomlarını belirledik.

Sonuç olarak

- Cebirsel formülasyon, interpolasyon polinomunu akla gelen en uygun

$$\left\{1, t, t^2, \ldots, t^n\right\}$$

lineer bağımsız kümesinin lineer kombinasyonu olarak ifade etmekte ve bu kombinasyonun katsayıları ise Vondermonde sisteminin çözümü yardımıyla elde edilmektedir. Interpolasyon verilerindeki değişiklik, çözülmesi gereken lineer sistemin boyutunu değiştirmekte ve işlemlerin tekrar edilmesini zorunlu kılmaktadır. Ayrıca çözülmesi gereken lineer sistemin katsayı matrisi olan Vondermonde matrisi ile kararlı değildir, yani oluşabilecek yuvarlama hataları işlem sonuçlarını olumsuz etkileyebilmektedir. Bu nedenle cebirsel yöntemin kullanılması tavsiye edilmemektedir.

- Newton formülasyonu ise interpolasyon polinomunu akıllıca seçilen

$$\{N_0(t), N_1(t), \ldots, N_n(t)\}$$

lineer bağımsız kümesinin uygun bir lineer kombinasyonu olarak ifade etmekte ve bu kombinasyonun katsayıları ise ardışık olarak hesaplanabilen

$$\{f_0, f[t_0, t_1], f[t_0, t_1, t_2], \ldots, f[t_0, t_1, \ldots, t_n]\}$$

bölünmüş farklarından oluşmaktadır. Interpolasyon verilerine yeni bir ilave ise polinoma kolayca ilave edilebilen sadece ekstra bir terimin ilavesini zorunlu kılmaktadır. Bu özellik uygulamalar açısından büyük bir avantaj olarak değerlendirilmektedir.

- Lagrange formülasyonu ise interpolasyon polinomunu

$$\{L_0(t), L_1(t), \ldots, L_n(t)\} \tag{4.18}$$

lineer bağımsız kümesinin bir lineer kombinasyonu olarak belirlemektedir. Söz konusu lineer bağımsız küme öyle belirlenmektedir ki lineer kombinasyonun katsayıları otomatik olarak verilen nokta çiftlerinin ordinatları, yani $y_i, i = 0, 1, \ldots, n$ sabitleri olmaktadır. Veri sayısındaki herhangi bir değişiklik ise lineer bağımsız kümenin her bir $L_i(t)$

elemanının önceki veri kümesi için elde edilenlerden bağımsız olarak yeniden hesaplanmasını gerektirmektedir. Bu özellik ise uygulamalar açısından büyük bir dezavantaj olarak değerlendirilmektedir.

- Cebirsel formülasyon ve Newton formülasyonunun uygulanmasında önemli bileşen lineer kombinasyondaki sabitlerin belirlenmesi iken, Lagrange yönteminde asıl sorun lineer kombinasyon için gerekli (4.18) lineer bağımsız polinomlar kümesinin belirlenmesidir.
- Cebirsel formülasyon ve Newton formülasyonuyla elde edilen interpolasyon polinomunun herhangi bir interpolasyon noktasındaki değerinin hesaplanması $n-$inci dereceden bir polinomunun sıradan herhangi bir noktadaki değerinin hesaplanması problemidir. Fakat aynı işlem Lagrange formülasyonunda her biri $n-$inci dereceden olan ve (4.18) da yer alan $n+1$ adet polinomun verilen interpolasyon noktasındaki değerlerinin hesaplanması, yani n kat daha fazla işlem gerektirmektedir.

4.3 İnterpolasyon işleminde hata

İnterpolasyon işleminde oluşan hata aşağıdaki teoremle ifade edilmektedir[1].

TEOREM 4.1. *(İnterpolasyon işleminde hata): Eğer $P_n(t)$ interpolasyon polinomu elde edilirken kullanılan y_i değerleri $y = f(t)$ bağıntısı ile tanımlı $(n+1)$-inci basamaktan türevi mevcut olan bir f fonksiyonunun t_i noktasındaki değerleri, yani*

$$y_i = f(t_i), i = 0, 1, ..., n$$

ve I ise tüm t_i noktalarını içeren en küçük bir açık aralık olmak üzere, her $t \epsilon I$ için

$$\begin{aligned} E(t) &: = f(t) - P_n(t) \\ &= (t-t_0)(t-t_1)...(t-t_n) f^{(n+1)}(\xi_t)/(n+1)! \end{aligned} \tag{4.19}$$

bağıntısı sağlanacak biçimde bir $\xi_t \in I$ noktası mevcuttur.

İspat.

Herhangi bir i için eğer $t = t_i$ ise

$$E(t_i) = P_n(t_i) - f(t_i) = 0$$

olduğundan (4.19) sağlanır. I açık aralığında seçilen(veya verilen) bir $t \neq t_i$ noktasını gözönüne alalım ve

$$w(x) = (x - t_0)(x - t_1)...(x - t_n) \tag{4.20}$$

olmak üzere

$$\beta = \frac{f(t) - P_n(t)}{w(t)}$$

sabitini tanımlayalım. (t seçilen(veya verilen) bir nokta olduğu için β sabittir). Ayrıca

$$\psi(x) = f(x) - P_n(x) - \beta w(x)$$

olarak tanımlayalım. $\psi(x)$ fonksiyonu aşağıdaki özellikleri sağlar:

- $\psi(t) = 0$
- $\psi(t_i) = 0, i = 0, 1, ..., n$
- $\psi \in C^{(n+1)}[a, b]$

ψ fonksiyonu $[a, b]$ de $n+2$ adet sıfıryerine sahiptir. Rolle teoreminden[7] ψ' fonksiyonu $[a, b]$ de $n + 1$ adet sıfıryerine sahiptir. İteratif olarak $\psi^{(n+1)}$ fonksiyonu $[a, b]$ de tek bir ξ sıfır yerine sahip olmalıdır:

$$\psi^{(n+1)}(\xi) = 0$$

ve

$$w^{(n+1)}(x) = (n + 1)!$$

den

$$\psi^{(n+1)}(\xi) = f^{(n+1)}(\xi) - \beta(n + 1)! = 0$$

veya

$$\frac{f^{(n+1)}(\xi)}{(n + 1)!} = \beta = \frac{f(t) - P_n(t)}{w(t)}$$

elde ederiz ki bu da ispatlanması istenen sonuçtur.

(4.19) ile tanımlanan $E(t)$ ye t noktasında oluşan interpolasyon hatası adı verilir.

- Herhangi bir i için $E(t_i) = 0$ olduğuna dikkat edelim. Diğer bir değimle, interpolasyon polinomu ve f fonksiyonu interpolasyon noktalarında birbirlerine eşittirler.

- Eğer f fonksiyonu özel olarak derecesi n den küçük veya n ye eşit bir polinom ise bu taktirde $f^{(n+1)}(t) \equiv 0$ olacağı için $E(t) \equiv 0$ olur, yani elde edilen interpolasyon polinomu, verilerin elde edildiği polinom fonksiyona eşit olur.

- (4.19) bağıntısı sayısal analiz işlemlerinde, özellikle de sayısal integrasyonda $f(t)$ yerine $P_n(t)$ kullanılması durumunda oluşan hatayı tahmin edebilmek için kullanılır.

- Yine (4.19) den yüksek mertebeden türevleri I açık aralığında sınırlı olan f fonksiyonları ile gerçekleştirilen interpolasyon işlemlerinde veri sayısının artmasıyla hatanın sıfıra yaklaşması gerektiği görülür.

- Eşit aralıklı, yani

$$t_{i+1} - t_i = h, i = 0, 1, ..., n-1$$

özelliğini sağlayan veriler yerine, 4.20 fonksiyonunun mutlak değerinin maximumunu gözönüne alınan aralıktaki minimize eden t_i lerin seçilmesi durumunda interpolasyon hatası da minimize edilmiş olur. Söz konusu değerlerin belirlenmesi teknik işlemler gerektirmektedir ve bu çalışma kapsamı dışındadır.

Alıştırmalar 4.1.

1. *Grafiği* $(1, -1), (2, 3)$ *noktalarından geçen polinomu*

   - *cebirsel yöntem,*
   - *Newton yöntemi ve*
   - *Lagrange yöntemi yardımıyla belirleyiniz.*

2. *Soru 1 de verilen noktalara ilaveten* $(3, 9)$ *noktasından da geçen polinomu yine her üç yöntem yardımıyla belirleyiniz. İnterpolasyon işleminde nokta ilavesi söz konusu ise hangi yöntemi kullanmayı tercih edersiniz? Genelde hangi yöntemi tercih edersiniz? Neden?*

3. *Faklı* t_i *düğüm noktaları ile oluşturulan Vandermond matrisinin determinantı sıfırdan farklı olması gerektiğini* 3×3 *lük*

$$\begin{bmatrix} 1 & t_0 & t_0^2 \\ 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \end{bmatrix}$$

matrisi için gösteriniz. Çarpım notasyonu yardımıyla determinantı ifade ediniz.

4. *Aşağıda verilen Newton bölünmüş fark tablosunda boş bırakılan yerleri doldurunuz.*

t_i	y_i			
1	1			
2	3	–		
3	5	–	–	
4	8	–	–	1/6

5. *Soru 4 de verilen bölünmüş fark tablosuna ait $P_3(t)$ Newton interpolasyon polinomunu belirleyiniz. $P_3(3/2)$ değerini Horner yöntemi yardımıyla hesaplayınız.*

6. *Grafiği $(0,1),(1,2),(2,4)$ noktalarından geçen interpolasyon polinomunu belirleyiniz.*

7. *Soru 6 daki noktalara ilaveten grafiği $(3,8)$ noktasından da geçen interpolasyon polinomunu Newton formülü yardımıyla belirleyiniz.*

8. *Beş adet nokta çiftinin üçüncü elemanında oluşacak olan ε hatasının Newton bölünmüş fark tablosuna nasıl yayıldığını inceleyelim: Bu amaçla $t_i = i, i = 0,1,2,3,4$ olmak üzere y_i gerçek değerleri ile*

$$Y_i = \begin{cases} y_i & i \neq 3 \\ y_i + \varepsilon & i = 3 \end{cases}$$

olmak üzere Y_i değerleri arasındaki farkları içeren

t	0	1	2	3	4
E	0	0	ε	0	0

veri kümesine ait Newton bölünmüş fark tablosunu hazırlayınız. Tablonun üst köşegeninde yer alan ε lu terimlere dikkat ediniz. Üçüncü bileşende oluşan hata tabloya nasıl yayılmaktadır?

9. *Grafiği*

t	0	1	2	3	4
y	1	2	4	8	12

noktalarından geçen $P_4(t)$ polinomunu ve

x	0	1	2	3	4
y	1	2	$4+\varepsilon$	8	12

noktalarından geçen $P_4(t,\varepsilon)$ polinomunu belirleyiniz. $P_4(t)$ ve $P_4(t,\varepsilon)$ polinomlarını karşılaştırınız. Üçüncü bileşende oluşan hata interpolasyon polinomunu nasıl etkilemektedir? Bu durumda hata içermesi muhtemel veri kümesi içerisinde bilinmeyen bir değeri interpolasyon polinomu yardımıyla tahmin etmek sağlıklı bir yaklaşım olur mu?

10. *$y = f(t) = t^2 + 1$ fonksiyonu verilsin.*

    - *$t = 0$ ve $t = 1$ noktalarında f fonksiyonuyla aynı değerleri alan $P_1(t)$ interpolasyon polinomunu belirleyiniz .*
    - *$f(t) - P_1(t)$ interpolasyon hatasını belirleyiniz.*
    - *(4.19) ile verilen $E(t)$ hatasını belirleyiniz. Elde ettiğiniz sonuç (b) deki sonuçla aynıdır, neden?*
    - *$[0, 1]$ aralığında $E(t)$ hata fonksiyonunun grafiğini çizerek, hatanın bu aralık içeresinde en fazla hangi noktada olduğunu gözlemleyiniz.*

11. *Fonksiyona t_0 noktası komşuluğunda $n-$inci dereceden Taylor polinomu ile yaklaşılması durumunda oluşan ve ilgili bölümde verilen hata formülü ile interpolasyon polinomu için verilen (4.19) hata formülünü karşılaştırınız. Taylor yaklaşımında hata, t_0 noktasından uzaklaşıldıkça artarken, interpolasyon polinomunda ise hatanın interpolasyon noktalarından uzaklaşıldıkça artacağını gözlemleyiniz.*

12. *$P_n(x)$ interpolasyon polinomları,* n *in artan değerleri için $[a, b]$ aralığında seçilen eşit aralıklı düğüm noktaları ile verilerin elde edildiği fonksiyona yakınsaması gerekmez. Carl Runge'ye ait aşağıdaki klasik örneği gözönüne alalım.*

$$f(x) = \frac{1}{1+x^2}, x \in [-5, 5]$$

Aşağıdaki düğüm noktalarına karşılık gelen interpolasyon polinomunu belirleyiniz.

- $x_0 = -5, x_1 = 0, x_2 = 5$

- $x_i = -5 + \frac{5}{2}i, i = 0, 1, 2, 3, 4$
- $x_i = -5 + \frac{5}{8}i, i = 0, 1, ..., 8$

13. *Elde ettiğiniz interpolasyon polinomları ve verilen $f(x)$ fonksiyonunun aynı eksende grafiğini çizerek, artan n düğüm nokta sayısı için $P_n(x)$ interpolasyon polinomunun f fonksiyonuna yakınsamadığını gözlemleyiniz.*

14. *Soru 12 yi*

$$f(x) = |x|$$

fonksiyonu için $[-1, 1]$ aralığında seçeceğiniz eşit aralıklı $3, 5, 7$ ve 9 adet düğüm noktası için tekrar ediniz. Aynı sonucu gözlemliyor musunuz? O halde interpolasyon polinomununun yakınsama problemi verilerin elde edildiği $f(x)$ fonksiyonunun türevlenebilirlik özelliği ile ilgili olmamalıdır.

15. *Soru 12 de verilen f fonksiyonu ve $a = -5$, $b = 5$ için*

$$x_k = \frac{1}{2}(a + b) + \frac{1}{2}(b - a)cos(\frac{2k - 1}{2n}\pi), k = 1, 2, ..., n$$

düğüm noktaları ile seçeceğiniz $n = 3, 5, 9$ değerleri için tekrar ediniz. Ne gözlemliyorsunuz?

16. *Bir fonksiyona ait süreklilik, düzgün süreklilik ve mutlak süreklilik kavramları arasında nasıl bir ilişki olduğunu araştırınız.*

17. *Bir fonksiyon dizisinin noktasal yakınsaklık ve düzgün yakınsaklık kavramları arasında nasıl bir ilişki olduğunu araştırınız.*

18. *Eğer f fonksiyonu $[a, b]$ aralığında mutlak sürekli ise, bu taktirde $[a, b]$ aralığına dönüştürmüş olan Chebyshev sıfıryerlerini(soru 15 te tanımlanan x_k noktaları) düğüm noktaları olarak kabul eden $P_n(x)$ interpolasyon polinomları artan n değerleri için f fonksiyonuna $[a, b]$ aralığında düzgün olarak yakınsar. Bu sonuç Soru 15 deki gözlemlerinizle uyumlu mudur?*

19. *Apsisleri farklı olan $(t_i, y_i), i = 0, 1, 2, ..., n$, nokta çifti verilmiş olsun. $p(t_i) = y_i$ özelliğini sağlayan interpolasyon polinomunun derecesinin n e eşit veya n den küçük olduğunu gösteriniz.*

20. *Soru 19 daki özellikleri sağlayan ve derecesi n den küçük veya n e eşit olan birden fazla polinomun olamayacağını ispatlayınız.*

Kaynaklar

[1] Atkinson, K. An Introduction to Numerical Analysis, John Wiley & Sons, 1988.

[2] Coşkun, E. OCTAVE ile Sayısal Hesaplama ve Kodlama(URL:erhancoskun.com.tr).

[3] Hildebrand, F. B., Introduction to Numerical Analysis, Dover Publications, Inc., 1987.

[4] Kincaid, D., Cheney, W., Numerical Analysis, Brooks/Cole, 1991.

[5] Mathews, J., Numerical Methods for Mathematics, Science and Engineering, Prentice-Hall, 1997.

[6] Stoer, J., Bulirsh, R., Introduction to Numerical Analsis, Springer-Verlag, 1976.

[7] S. W., Warren, Zill, D. G., Calculus: Early Transcendentals, Çeviri: Matematik Cilt I, II(Çeviri editörü İsmail Naci Cangül), Nobel Akademik Yayıncılık, 2010.

Bölüm 5

Elemanter Fonksiyonlarla Yaklaşım ve Hata

Bu bölümde öncelikle verilen bir ayrık veri kümesi için

- standart ve
- ağırlıklı en küçük kareler yöntemi ile en uygun yaklaşım polinomunun nasıl belirleneceğini inceliyoruz. Ayrıca
- bir aralık üzerinde verilen herhangi bir integrallenebilir fonksiyona daha basit fonksiyonlarla uygun yaklaşımların nasıl geliştirilebileceğini MATLAB/Octave'ın **vektör cebirsel** işlem yetenekleri yardımıyla inceliyoruz.

Konunun **skaler cebirsel** analizi için [1],[4] ve [9] temel referans kaynaklarını öneririz.

5.1 Giriş

Herhangi bir araştırma sonucunde belirli bir amaca yönelik olarak elde edilmiş olan ve apsisleri birbirlerinden farklı olan

$$A = \{(x_i, y_i) | i = 0, 1, ..., n\}$$

nokta çiftleri kümesinin verildiğini kabul edelim. Önceki bölümde

$$P(x_i) = y_i, i = 0, 1, ..., n$$

özelliğini sağlayan ve derecesi $\leq n$ olan interpolasyon polinomunun nasıl elde edildiğini öğrendik. Yine önceki bölümde gözlemlediğiniz üzere, verilerde hata olması durumunda söz konusu hata interpolasyon polinomunun yüksek dereceli terimlerinin katsayılarını etkiler ve böylece hatalı verilerle elde edilen interpolasyon polinomu ile gerçekleştirilen interpolasyon işlemi de güvenilirliğini kaybeder. Bu durumda yüksek dereceli polinomlarla interpolasyon yerine, verilere uygun fonksiyonlarla yaklaşım gerçekleştirilerek, bilinmeyen noktadaki değer tahmini elde edilen yaklaşım fonksiyonu ile gerçekleştirilir.

5.2 Ayrık veri kümesine elemanter fonksiyonlarla yaklaşım

Hata içeren noktalardan geçen yüksek dereceli interpolasyon polinomunu bulmak yerine, hatalı veri kümesine yeterince yakın noktalardan geçen daha düşük dereceli polinomu belirleyerek interpolasyon işlemini elde edilen düşük dereceli polinomla gerçekleştirmek daha akılcı bir yöntemdir.

Standart En Küçük Kareler Yöntemi ile Yaklaşım

Veri kümemizin

$$(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$$

nokta çiftlerinden oluştuğunu ve bu nokta çiftlerine en yakın

$$P_1(x) = a + bx$$

polinomunu belirlemek istediğimizi kabul edelim. Burada a ve b bilinmeyenlerini, veri kümesinin $P_1(x)$ polinomuna yakınlığının "uygun bir ölçüsünü" minimize edecek biçimde belirlemek istiyoruz. Bu durumda en uygun ölçü iki normu yardımıyla

$$E(a, b) = \sum_{i=1}^{m} (P_1(x_i) - y_i)^2 = \sum_{i=1}^{m} (a + bx_i - y_i)^2 \tag{5.1}$$

ile tanımlanmaktadır. Amacımız $E(a, b)$ yi minimum yapan a ve b değerlerini belirlemektir. Sözkonusu minimum nokta için gerek şart (yeter şart değil!)

$$\frac{\partial E}{\partial a} = 0, \frac{\partial E}{\partial b} = 0$$

eşitliklerinin sağlanmasıdır. Fakat

$$\begin{aligned} \frac{\partial E}{\partial a} &= 2\sum_{i=1}^{m}(a+bx_i-y_i)=0 \\ &\Rightarrow \quad ma+\left(\sum_{i=1}^{m}x_i\right)b=\sum_{i=1}^{m}y_i \end{aligned} \tag{5.2}$$

ve

$$\begin{aligned} \frac{\partial E}{\partial b} &= 2\sum_{i=1}^{m}(a+bx_i-y_i)x_i=0 \\ &\Rightarrow \quad \left(\sum_{i=1}^{m}x_i\right)a+\left(\sum_{i=1}^{m}x_i^2\right)b=\sum_{i=1}^{m}x_iy_i \end{aligned} \tag{5.3}$$

elde ederiz. (5.2) ve (5.3) sistemi çözülerek a ve b değerleri ve dolayısıyla da istenilen $P_1(x)$ polinomu elde edilmiş olunur. (5.1) deki karelerin toplamını minimize etmek(en küçük yapmak) için kullanılan bu yönteme, En Küçük Kareler Yöntemi(EKKY) adı verilir.

Öte yandan (5.2) ve (5.3 sistemi matris-vektör noktasyonu yardımıyla da ifade edilebilir: Öncelikle

$$\begin{aligned} \mathbf{1} &= [1,1,\cdots,1]^T, \\ \mathbf{x} &= [x_1,x_2,\cdots,x_m]^T, \\ \mathbf{y} &= [y_1,y_2,\cdots,y_m]^T, \\ \mathbf{u} &= [a,b]^T \end{aligned}$$

vektörlerini ve

$$A=[\mathbf{1}\ \mathbf{x}]_{m\times 2}$$

matrisini tanımlayalım. Bu durumda

$$A^TA=\begin{bmatrix} m & \sum_{i=1}^{m}x_i \\ \sum_{i=1}^{m}x_i & \sum_{i=1}^{m}x_i^2 \end{bmatrix}, A^T\mathbf{y}=\begin{bmatrix} \sum_{i=1}^{m}y_i \\ \sum_{i=1}^{m}x_iy_i \end{bmatrix}$$

olup, (5.2) ve (5.3) sistemi

$$A^TA\mathbf{u}=A^T\mathbf{y} \tag{5.4}$$

olarak ifade edilebilir.

ÖRNEK 5.1. $(0,0),(1,3/2),(2,1/2),(3,4),(4,3)$ *noktaları için*

$$P_1(x) = a + bx$$

biçimindeki birinci dereceden en iyi yaklaşım polinomunu En Küçük Kareler Yöntemini kullanarak belirleyiniz.

Çözüm.

Örneğimiz için

$$A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \end{bmatrix}, y = \begin{bmatrix} 0 \\ 3/2 \\ 1/2 \\ 4 \\ 3 \end{bmatrix}$$

ile

$$A^T A = \begin{bmatrix} 5 & 10 \\ 10 & 30 \end{bmatrix}, A^T y = \begin{bmatrix} 10 \\ 53/2 \end{bmatrix}$$

olup, bilinmeyen vektörü $u = [a\ b]^T$ olmak üzere, (5.4) den

$$\begin{aligned} 5a + 10b &= 10 \\ 10a + 30b &= 53/2 \end{aligned}$$

denklem sistemini elde ederiz. Bu sistemi çözerek,

$$a = 1/10; b = 17/20$$

elde ederiz. Elde edilen

$$P_1(x) = 1/10 + 17/20x$$

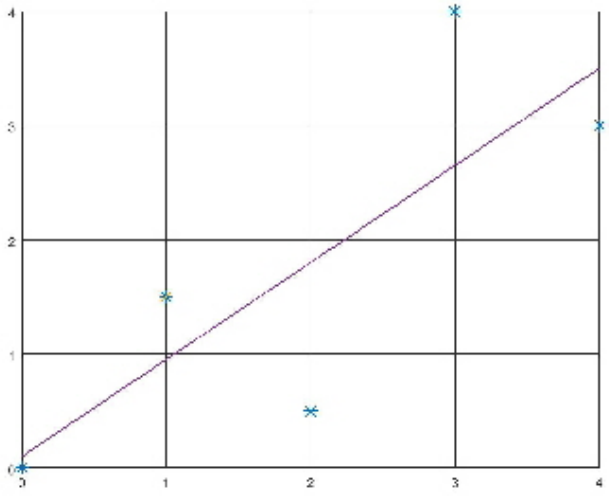
doğrusunun $(*)$ ile belirtilen verilere uygun mesafelerden geçerek (5.1) ile verilen $E(a,b)$ hatasını minimize etmeye çalıştığı görülmektedir.

ÖRNEK 5.2. *Verilen* $(x_i, y_i), i = 1, 2, \cdots, m$ *noktalarına uygun* $P_1(x) = a + bx$ *polinomunu belirleyerek, (5.1) hatasını belirledikten sonra*

$$t \in (\min_{1 \le i \le m}(x_i), \max_{1 \le i \le m}(x_i))$$

için t *noktasındaki değeri* $P_1(x)$ *yardımıyla tahmin*

$$[toplam\_hata, tahmin] = ekky1(x, y, t)$$



Şekil 5.1: Örnek 5.1 nokta çiftleri($*$) ve $P_1(x)$ yaklaşım polinomu

komutu ile çalışan bir En Küçük Kareler Yöntemi uygulaması geliştiriniz. Burada x ve y sırasıyla nokta çiftlerinin apsis ve ordinatlarını içeren vektörlerdir.

Çözüm.

Probleme ait yöntemimiz En Küçük Kareler Yöntemi olacağından, öncelikle yönteme ait algoritmayı geliştirelim:

Algoritma 5.1. 1. *Girdi*(x, y)

2. $m := x$ *in eleman sayısı*

3. $x := x^T, y := y^T$ *sütun vektörini tanımla.*

4. *birler isimli* m *bileşenli 1 rakamlarından oluşan sütun vektörünü tanımla.*

5. $A := [birler\ x]$ *matrisini oluştur.*

6. $B := A^T A$ *matrisi ve* $c := A^T y$ *vektörünü oluştur*

7. $Bu = c$ *sistemini çözerek* $u := [a\ b]$ *bilinmeyenlerini belirle*

8. *(x, y) ikililerinin eksende yerlerini işaretle*

9. *$p(x) = a + bx$ polinomunun grafiğini aynı eksende çizdir.*

10. *a ve b değerlerini geri gönder*

Yukarıdaki algoritmaya ait Program (5.1) aşağıda verilmektedir.

```
function [a,b]=ekky1(x,y)
% (x,y) nokları ile uyumlu P(x)=a+bx
% polinomunu EKKY ile hesaplar
% polinom ve nokta ciftlerini grafigini cizer
%-------------------------------------------

x=x';y=y';m=length(x);
birler=ones(m,1);
A=[birler x];
B=A'*A;
c=A'*y;
u=B\c;
p=@(xx) u(1)+u(2)*xx;
xx=x(1):0.1:x(end);
plot(x,y,'*','markersize',10); hold on;
yy=p(xx); plot(xx,yy);
a=u(1);b=u(2);
```

Program 5.1: EKKY uygulaması

Test
>> x=[0 1 2 3 4];
>> y=[0 3/2 1/2 4 3];
>> [a,b]=ekky1(x,y)
>>
a = 1/10
b = 17/20
Eğer

$$P_2(x) = a + bx + cx^2$$

biçiminde olup, veri kümesine en iyi yaklaşan ikinci dereceden polinomu belirlemek istersek yine aynı işlemleri takip ederiz, ancak bu defa çözülmesi gereken lineer sistem 3×3 lük bir sistem olur. Bu durumda A matrisi

$$A = \begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ \vdots & \vdots & \vdots \\ 1 & x_m & x_m^2 \end{bmatrix}$$

olarak tanımlanmalıdır.

Veri kümesi üstel veya logaritmik bir dağılıma sahip olması durumunda en iyi yaklaşım polinomu da veri kümesi için iyi bir yaklaşım olmaz. Bir deney sonucunda değerleri zamanla üstel olarak artan veya azalan pozitif ordinatlı bir veri kümesi için en iyi yaklaşım

$$y = ae^{bx}, a > 0$$

biçiminde olmalıdır. Bu durumda

$$E(a,b) = \sum_{i=1}^{m} (ae^{bx_i} - y_i)^2 \tag{5.5}$$

ifadesini minimize eden a ve b değerlerinin belirlenmesi gerekir. Ancak bu durumda

$$\begin{aligned} \frac{\partial E}{\partial a} &= 2\sum_{i=1}^{m} (ae^{bx_i} - y_i)e^{bx_i} = 0, \\ \frac{\partial E}{\partial b} &= 2\sum_{i=1}^{m} (ae^{bx_i} - y_i)ax_i e^{bx_i} = 0 \end{aligned}$$

biçimde yazılabilen a ve b bilinmeyenleri için nonlineer bir sistem elde ederiz ki bu sistemin çözümü de Newton yöntemi gibi sayısal bir yöntem gerektirir.

Alternatif bir yöntem takip edebiliriz:

$$y(x_i) = ae^{bx_i} (a > 0)$$

değerlerini verilen $y_i > 0$ değerlerine yaklaştırmaya çalışmak $\ln(y(x_i))$ değerlerini $\ln(y_i)$ değerlerine yaklaştırmaya denktir: Gerçekten de

$$y(x_i) \to y_i$$

ise, logaritma fonksiyonunun sürekliliği gereği

$$\ln(y(x_i)) \to \ln(y_i)$$

dir. Tersine

$$\ln(y(x_i)) \to \ln(y_i)$$

ise

$$\ln(y(x_i)) - \ln(y_i) \to 0$$

veya

$$\ln(\frac{y(x_i)}{y_i}) \to 0 \Rightarrow \frac{y(x_i)}{y_i} \to 1,$$

yani $y(x_i) \to y_i$ dir.

O halde (5.5) ile verilen fonksiyonu minimize etme problemi

$$\hat{E}(a,b) = \sum_{i=1}^{m}(\ln(ae^{bx_i}) - \ln(y_i))^2 \tag{5.6}$$

$$= \sum_{i=1}^{m}(\ln(a) + bx_i - \ln(y_i))^2 \tag{5.7}$$

fonksiyonunu minimize etme problemine denktir. $\hat{a} = \ln(a)$,ve

$$\hat{y}_i = \ln(y_i), i = 1, 2, ..., m$$

olarak tanımlarsak, problem (5.1) a benzer olarak

$$\hat{E}(\hat{a},b) = \sum_{i=1}^{m}(\hat{a} + bx_i - \hat{y}_i)^2 \tag{5.8}$$

ifadesini minimize eden $\hat{a}$ ve b değerlerini belirleme problemine, diğer bir değimle

$$\hat{y} = \hat{a} + bx$$

lineer ifadesini elde etme problemine dönüşür. Yukarıda gerçekleştirilen işleme EKKY için **lineerleştirme işlemi** adı verilmektedir.

ÖRNEK 5.3. $(0, 1/2), (1, 2), (2, 5), (3, 8)$ *noktaları için*

$$y = ae^{bx}$$

biçimindeki en iyi yaklaşımı EKKY ile belirleyiniz.

Çözüm.

(5.2) ve (5.3) sisteminin katsayılarını elde etmek için aşağıdaki tabloyu oluşturalım:

y_i	$\hat{y}_i = \ln(y_i)$
$1/2$	$\ln(1/2) = -\ln 2$
2	$\ln(2)$
5	$\ln(5)$
8	$\ln(8) = 3\ln(2)$

Buna göre

$$A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{bmatrix}, \hat{y} = \begin{bmatrix} -\ln(2) \\ \ln(2) \\ \ln(5) \\ 3\ln(2) \end{bmatrix}$$

için

$$A^T A = \begin{bmatrix} 4 & 6 \\ 6 & 14 \end{bmatrix}, A^T \hat{y} = \begin{bmatrix} 3\ln(2) + \ln(5) \\ 10\ln(2) + 2\ln(5) \end{bmatrix}$$

$$\begin{aligned} 4\hat{a} + 6b &= 3\ln(2) + \ln(5) = 3.6889 \\ 6\hat{a} + 14b &= 10\ln(2) + 2\ln(5) = 10.15 \end{aligned}$$

olarak elde ederiz. Bu sistemi çözerek

$$\hat{a} \doteq -0.4628, b \doteq 0.9233$$

elde ederiz. O halde

$$a = e^{\hat{a}} = 0.6295$$

olmak üzere,

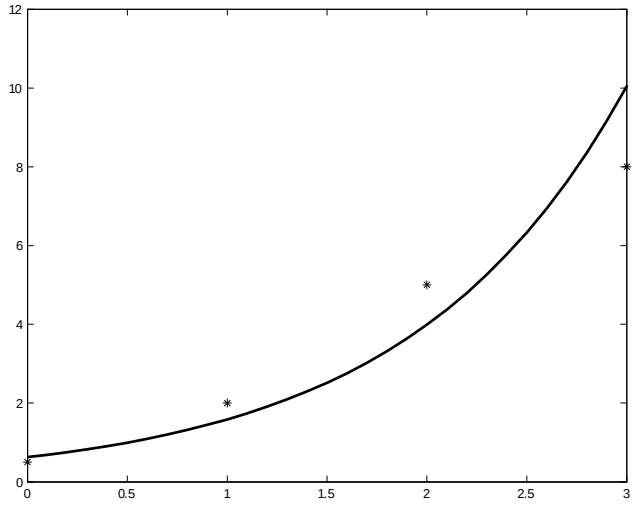
$$y = ae^{bx} = 0.6295e^{0.9233x}$$

elde ederiz. Verilen nokta çiftleri ve elde edilen eğrinin grafiği Şekil 5.2 de verilmektedir.

Benzer biçimde (x_i, y_i) nokta çiftleri için

$y = \frac{1}{a+bx}$	biçiminde eğri aranıyorsa	$\hat{y} = \frac{1}{y}$
$y = \frac{1}{\sqrt{a+bx}}$	"	$\hat{y} = \frac{1}{y^2}$

biçiminde dönüşümlerle EKKY problemi lineer probleme dönüştürülür.



Şekil 5.2: Elde edilen üstel fonksiyon ve verilen nokta çiftleri $(*)$

Ağırlıklı En Küçük Kareler Yöntemi ile Yaklaşım

Ağırlıklı En Küçük Kareler yöntemi ile verilen veri kümesi için uygun yaklaşım belirlenirken, verilerin güvenilirliği bilgisi de dikkate alınır. Bu amaçla

$$w = (w_1, w_2, \cdots, w_m), w_i \geq 0, \sum_{i=1}^{m} w_i = 1$$

özelliğini sağlayan w_i çarpanları(veya ağırlıkları) için

$$E(a, b; w) = \sum_{i=1}^{m} w_i (ax_i + b - y_i)^2 \tag{5.9}$$

ile tanımlanan normu minimize eden a ve b sabitleri belirlenir. Sözkonusu minimum nokta için gerek şart (yeter şart değil!)

$$\frac{\partial E(a, b; w)}{\partial a} = 0, \frac{\partial E(a, b; w)}{\partial b} = 0$$

eşitliklerinin sağlanmasıdır. Fakat

$$\begin{aligned}\frac{\partial E}{\partial a} &= 2\sum_{i=1}^{m} w_i(a+bx_i-y_i)=0 \\ &\Rightarrow \left(\sum_{i=1}^{m} w_i\right)a+\left(\sum_{i=1}^{m} x_i w_i\right)b=\sum_{i=1}^{m} y_i w_i \qquad (5.10)\end{aligned}$$

ve

$$\begin{aligned}\frac{\partial E}{\partial b} &= 2\sum_{i=1}^{m} w_i(a+bx_i-y_i)x_i=0 \\ &\Rightarrow \left(\sum_{i=1}^{m} x_i w_i\right)a+\left(\sum_{i=1}^{m} x_i^2 w_i\right)b=\sum_{i=1}^{m} x_i y_i w_i \qquad (5.11)\end{aligned}$$

elde ederiz. (5.10) ve (5.11) sistemi çözülerek a ve b değerleri ve dolayısıyla da istenilen $P_1(x)$ polinomu elde edilmiş olunur. (5.9) daki karelerin toplamını minimize etmek(en küçük yapmak) için kullanılan bu yönteme, Ağırlıklı En Küçük Kareler Yöntemi(EKKY) adı verilir.

Öte yandan (5.10) ve (5.11) sistemi matris-vektör notasyonu yardımıyla da ifade edilebilir: Öncelikle

$$\begin{aligned}\mathbf{1} &= [1,1,\cdots,1]^T, \\ \mathbf{x} &= [x_1,x_2,\cdots,x_m]^T, \\ \mathbf{y} &= [y_1,y_2,\cdots,y_m]^T, \\ \mathbf{u} &= [a,b]^T\end{aligned}$$

vektörlerini ve

$$\begin{aligned}A &= [\mathbf{1}\ \mathbf{x}]_{m\times 2} \\ W &= \begin{bmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & w_m \end{bmatrix}\end{aligned}$$

matrisini tanımlayalım. Bu durumda

$$A^TWA=\begin{bmatrix} \sum_{i=1}^m w_i & \sum_{i=1}^m x_i w_i \\ \sum_{i=1}^m x_i w_i & \sum_{i=1}^m x_i^2 w_i \end{bmatrix}, A^TW\mathbf{y}=\begin{bmatrix} \sum_{i=1}^m x_i y_i \\ \sum_{i=1}^m x_i y_i w_i \end{bmatrix}$$

olarak elde ederiz. İstenilen $\mathbf{u} = [a, b]^T$ vektörü ise

$$A^T W A\mathbf{u} = A^T W\mathbf{y}$$

sisteminin çözümü olarak elde edilir.

ÖRNEK 5.4. *Sırasıyla* $w_1 = 1/8, w_2 = 1/8$ *ve* $w_3 = 3/4$ *ağırlıklara sahip* $(0,0), (1,1/2), (2,4)$ *noktalar kümesini göz önüne alalım. Bu küme için en uygun* $P_1(x) = a + bx$ *polinomunu*

- Standart EKKY ve
- Ağırlıklı EKKY ile belirleyiniz

Çözüm.

$$A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \end{bmatrix}, W = \begin{bmatrix} 1/8 & 0 & 0 \\ 0 & 1/8 & 0 \\ 0 & 0 & 3/4 \end{bmatrix}, y = \begin{bmatrix} 0 \\ 1/2 \\ 4 \end{bmatrix}$$

olmak üzere Standart EKKY ile,

$$A^T A = \begin{bmatrix} 3 & 3 \\ 3 & 5 \end{bmatrix}, A^T y = \begin{bmatrix} 9/2 \\ 17/2 \end{bmatrix},$$

olmak üzere
$A^T A\mathbf{u} = A^T\mathbf{y}$ veya açıkça

$$\begin{aligned} 3a + 3b &= 9/2 \\ 3a + 5b &= 17/2 \end{aligned}$$

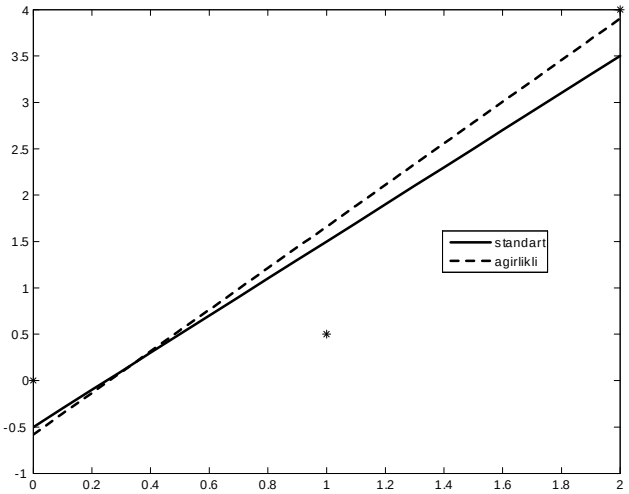
sistemini çözerek $a = -1/2, b = 2$ elde ederiz.
Ağırlıklı EKKY ile

$$A^T W A = \begin{bmatrix} 1 & 13/8 \\ 13/8 & 25/8 \end{bmatrix}, A^T W y = \begin{bmatrix} 49/16 \\ 97/16 \end{bmatrix}$$

solmak üzere

$$A^T W A\mathbf{u} = A^T W\mathbf{y}$$

sitemini çözerek, $a = -18/31, b = 139/62$ elde ederiz.
Ağırlıklı EKKY ile $w_3 = 3/4$ ağırlığına sahip noktaya daha yakınız!



Şekil 5.3: (x_i, y_i) noktaları(*) ile Standart ve Ağırlıklı EKKY polinom grafikleri

5.3 İntegrallenebilir fonksiyona elemanter fonksiyonlarla yaklaşım

Önceki bölümümde incelenen ayrık veri kümesi yerine, bazen bir $[c, d]$ aralığı üzerinde integrallenebilir bir f fonksiyonuna yüksek dereceli bir interpolasyon polinomu yerine daha basit fonksiyon veya polinomlar yardımıyla yaklaşım elde edilmek istenebilir. Verilen fonksiyona

$$P_1(x) = a + bx$$

gibi *en küçük kareler yaklaşım polinomunu* belirlemek için bu defa (5.1) deki toplam yerine integral sembolü yardımıyla polinom ve fonksiyon arasındaki farkın bir ölçüsü olan(L_2 normunun karesi)

$$\begin{aligned} E(a,b) &= \int_c^d (P_1(x) - f(x))^2 dx \\ &= \int_c^d (a + bx - f(x))^2 dx \end{aligned}$$

ifadesini minimum yapan a ve b değerlerini buluruz. Bu amaçla yine

$$\frac{\partial E}{\partial a} = 0, \frac{\partial E}{\partial b} = 0$$

sağlanmalıdır. Yani

$$\begin{aligned}\frac{\partial E}{\partial a} &= 2\int_c^d (a+bx-f(x))dx = 0\\ \frac{\partial E}{\partial b} &= 2\int_c^d (a+bx-f(x))xdx = 0\end{aligned}$$

veya daha açık olarak ifade edilen

$$\begin{aligned}\left(\int_c^d dx\right)a + \left(\int_c^d xdx\right)b &= \int_c^d f(x)dx\\ \left(\int_c^d xdx\right)a + \left(\int_c^d x^2dx\right)b &= \int_c^d xf(x)dx\end{aligned} \tag{5.12}$$

lineer denklem sistemini sağlayan a ve b değerleri verilen f fonksiyonuna $P_1(x)$ ile gösterilen en iyi yaklaşım polinomunu verir.

ÖRNEK 5.5. $f(x)=\sin(x)$ *fonksiyonuna* $[0,\pi/2]$ *aralığında*

$$P_1(x) = a+bx$$

biçimindeki en küçük kareler yaklaşım polinomunu belirleyiniz.

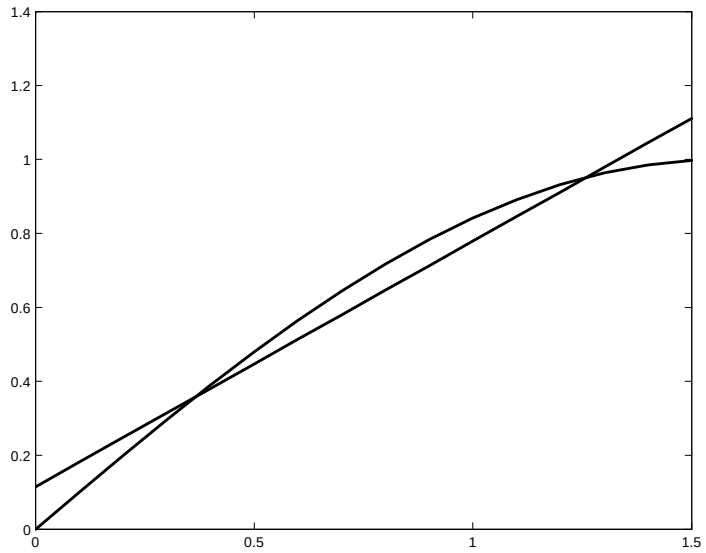
Çözüm.

$$E(a,b) = \int_0^{\pi/2}(a+bx-\sin(x))^2dx$$

fonksiyonelini minimize eden a ve b değerleri (5.12) den

$$\begin{aligned}\left(\int_0^{\pi/2} dx\right)a + \left(\int_0^{\pi/2} xdx\right)b &= \int_0^{\pi/2}\sin(x)dx\\ \int_0^{\pi/2}(xdx)\,a + \left(\int_0^{\pi/2} x^2dx\right)b &= \int_0^{\pi/2} x\sin(x)dx\end{aligned}$$

$$\int_0^{\pi/2}\sin(x)dx = 1 \text{ ve } \int_0^{\pi/2} x\sin(x)dx = -\int_0^{\frac{1}{2}\pi}(-\cos x)\,dx = 1$$



Şekil 5.4: $\sin(x)$ fonksiyonu ve $[0, 1.5]$ aralığındaki en küçük kareler yaklaşımı.

integral değerleri yardımıyla

$$\begin{aligned} \frac{\pi}{2}a + \frac{\pi^2}{8}b &= 1 \\ \frac{1}{8}\pi^2 a + \frac{1}{24}\pi^3 b &= 1 \end{aligned}$$

lineer denklem sisteminden

$$a = \frac{1}{\pi^2}\left(8\pi - 24\right) = 0.114\,77$$

ve

$$b = -\frac{1}{\pi^3}\left(24\pi - 96\right) = 0.664\,44$$

elde ederiz. O halde aradığımız en iyi yaklaşım polinomu

$$P_1(x) = 0.114\,77 + 0.664\,44x$$

dir.

$\sin(x)$ fonksiyonu ve en küçük kareler anlamındaki en iyi

$$P_1(x) = 0.114\,77 + 0.664\,44x$$

yaklaşım polinomunun grafiği Şekil 5.4 te verilmektedir.

Alıştırmalar 5.1.

1. $(0,1),(1,3),(2,1),(3,6)$ *veri kümesi için EKKY ile aşağıdaki biçimde belirtilen en iyi yaklaşım polinomlarını belirleyiniz.*

   (a) $P_0(x) = a$
   (b) $P_1(x) = a + bx$
   (c) $P_2(x) = a + bx + cx^2$
   (d) $P_3(x) = a + bx + cx^2 + dx^3$

2. *Soru 1 için verilen nokta kümesi ile elde ettiğiniz polinom grafiklerini aynı eksende gösteriniz.*

3. *Soru 1 için elde ettiğiniz her bir yaklaşım için (5.1) ile verilen hatayı hesaplayınız. Hata hangi yaklaşım için daha küçüktür?*

4. *Soru 1 i sırasıyla* $w_1 = 1/16, w_2 = 1/16, w_3 = 1/8, w_4 = 3/4$ *ağırlıkları ile ve Ağırlıklı EKKY ile tekrar ediniz.*

5. *Soru 1 de verilen nokta kümesi ve soru 4 te elde ettiğiniz polinom grafiklerini aynı eksende gösteriniz.*

6. *Soru 2 ve soru 5 te elde ettiğiniz grafikleri inceleyerek,* $w_i, i = 1,2,3,4$ *ağırlıklarının, yaklaşım polinomları üzerindeki etkisini inceleyiniz.*

7. *Grafiği Soru 1 de verilen noktalardan geçen* $Q_3(x)$ *interpolasyon polinomunu belirleyiniz.* $Q_3(x)$ *ile Soru 1(d) de elde ettiğiniz* $P_3(x)$ i *karşılaştırınız. Ne gözlemliyorsunuz?*

8. $(0,1),(1,2),(2,2.5),(3,2.8)$ *veri kümesi için EKKY e göre en iyi*

$$f(x) = \sqrt{a + bx}$$

   yaklaşım fonksiyonunu belirleyiniz ve oluşan $E(a,b)$ *hatasını hesaplayınız. Fonksiyon ve veri kümesinin grafiğini aynı eksende çizdiriniz.*

9. *Soru 8 de verilen veri kümesi için en iyi*

$$g(x) = \ln(a + bx)$$

   fonksiyonunu ve oluşan $E(a,b)$ *hatasını hesaplayınız. Elde ettiğiniz hatayı soru 8 deki değer ile karşılaştırınız. Belirtilen veri kümesi için* f *ve* g *fonksiyonlarından hangisi daha uygundur?*

10. $(0, 1/2), (2, 3), (3, 8)$ *veri kümesi için EKKY e göre en iyi*

$$f(x) = ae^{bx}$$

yaklaşım fonksiyonunu belirleyiniz.

11. $f(x) = \cos(x)$ *fonksiyonu için* $[-1, 1]$ *aralığında EKKY göre aşağıdaki biçimde belirtilen en iyi yaklaşım fonksiyonlarını ve oluşan*

$$E = \int_{-1}^{1} (f(x) - g(x))^2 dx$$

hatalarını belirleyiniz.

(a) $g(x) = a$

(b) $g(x) = a + bx$

(c) $g(x) = a + bx + cx^2$

12. Soru 11 de elde verilen f *için* $x = 0$ *noktasında sırasıyla sıfırıncı ve ikinci dereceden Taylor yaklaşımları sonucunda oluşan*

(a) $E = \int_{-1}^{1} (\cos(x) - 1)^2 dx$,

(b) $E = \int_{-1}^{1} (\cos(x) - (1 - x^2/2))^2 dx$

hatalarını elde ediniz. (a) ve (b) de elde ettiğiniz sonuçları sırasıyla, soru 11(a) ve soru 11(c) ile karşılaştırınız. Taylor mu yoksa EKKY yaklaşımı mı daha küçük hata üretmektedir?

13. Soru 11 ve 12 deki işlemleri Maxima ortamında da gerçekleştiriniz.

14. (x, y) *veri kümesi için en iyi* $n-$*inci dereceden polinomu hesaplayarak katsayılarını geri gönderen ve*

$$katsayi = ekkypol(x, y, n)$$

yazılımı ile çalışan bir MATLAB/Octave programı hazırlayınız. n *sayısının veri kümesindeki nokta sayısından büyük olmamasına dikkat ediniz. Hazırladığınız program* (x, y) *nokta çiftlerinin ve elde edilen polinomun grafiğini de aynı eksende çizmelidir.*

15. (x, y) *veri kümesi için en iyi*

$$y = ae^{bx}$$

fonksiyonunu hesaplayarak a *ve* b *değerlerini geri gönderen ve*

$$[a, b] = ekkyustel(x, y)$$

yazılımı ile çalışan bir MATLAB/Octave programı hazırlayınız. Hazırladığınız program (x, y) *nokta çiftlerini ve elde edilen*

$$y = ae^{bx}$$

fonksiyonunun grafiğini aynı eksende göstermelidir.

Kaynaklar

[1] Atkinson, K. An Introduction to Numerical Analysis, John Wiley & Sons, 1988.

[2] Coşkun, E. OCTAVE ile Sayısal Hesaplama ve Kodlama(URL:aves.ktu.edu.tr/erhan/dokumanlar).

[3] Coşkun, E. Maxima ile Sembolik Hesaplama ve Kodlama(URL:aves.ktu.edu.tr/erhan/dokumanlar).

[4] Kincaid, D., Cheney, W., Numerical Analysis, Brooks/Cole, 1991.

[5] Mathews, J., Numerical Methods for Mathematics, Science and Engineering, Prentice-Hall, 1997.

[6] MATLAB, Mathworks(*URL:mathworks.com).*

[7] Maxima, GNU özgür yazılım(*URL:maxima.sourceforge.net*).

[8] OCTAVE, GNU özgür yazılım(*URL:OCTAVE.sourceforge.net*).

[9] Stoer, J., Bulirsh, R., Introduction to Numerical Analysis, Springer-Verlag, 1976.

[10] S. W., Warren, Zill, D. G., Calculus: Early Transcendentals, Çeviri: Matematik Cilt I, II(Çeviri editörü İsmail Naci Cangül), Nobel Akademik Yayıncılık, 2010.

Bölüm 6

Nonlineer cebirsel denklemler

Uygulamalı bilimlerde çok sayıda problem, cebirsel bir fonksiyonun veya sistemin sıfıryeri veya sıfıryerlerinin bulunmasını gerektirir. Bu bölümde

- tek değişkenli ve reel değerli fonksiyonların sıfır yerlerini belirlemek amacıyla kullanılan sabit nokta iterasyon yöntemi,
- özel bir sabit nokta iterasyon yöntemi olan *Newton* yöntemi ve
- *Newton* yöntemindeki türeve sonlu fark yaklaşımı ile elde edilen kiriş yöntemini inceliyoruz. Ayrıca
- nonlineer cebirsel sistemler için *Newton* yöntemi,
- aralık üzerindeki tüm sıfıryerlerini belirlemek amacıyla ikiye bölme yöntemi ve
- r yarıçaplı disk içerisindeki tüm reel ve kompleks kökleri bulacak biçimde *Newton* yönteminin vektörel versiyonunu geliştirerek inceliyoruz.

Öncelikle sıfır yeri belirlenmesini gerektiren bazı problemlere göz atalım.

6.1 Pratik sıfır yeri problemleri

Fonksiyon sıfır yerlerinin ve özellikle de polinom sıfır yerlerinin(köklerinin) belirlenmesi problemi matematik başta olmak üzere bir çok alanda güncelliğini koruyan problemdir.

Özel olarak polinom sıfır yerlerinin belirlenebilmesi için çok sayıda çalışma gerçekleştirilmiş, derecesi üç ve dört olan polinomlar için *Cardano*[1] ya atfedilen ve pratik olarak kullanılamayacak kadar karmaşık formüller geliştirilmiştir. Ancak daha yüksek dereceli polinomların sıfır yerleri için herhangi bir formül geliştirilemeyeceği *Abel* ve Galois(Galoa okunur)' nın çalışmaları ile gösterilmiştir.

Sıfır yeri belirleme problemi değişik alanlarda sıkça karşılaşılan bir problemdir:

- Bir f fonksiyonunun grafiğinin x eksenini kesim noktaları, ekstremum noktaları veya büküm noktalarının belirlenmesi, matematiksel açıdan olduğu kadar ilgili fonksiyonun temsil ettiği pratik değerin analizi açısından da oldukça önemlidir. Bu bilgiler yardımıyla verilen fonksiyonun veya temsil ettiği pratik değerin belirli bir aralıktaki davranışını tahmin edebiliriz.

- Ekonomide bir ürünün x adetinin üretimi sonucunda oluşan ve $C(x)$ ile gösterilen maliyet ile ürünün satışından elde edilen $R(x)$ gelir fonksiyonları yardımıyla tanımlanan $f(x) = R(x) - C(x)$ fonksiyonunun sıfır yeri, gelirin maliyeti karşıladığı üretim miktarını verir ki bu değerin belirlenmesi oldukça önemlidir.

- $dx/dt = f(x)$ diferensiyel denkleminin denge noktaları(eğer mevcutsa) f fonksiyonunun reel sıfır yerleridir.

- $$\begin{aligned} dx/dt &= f(x,y) \\ dy/dt &= g(x,y) \end{aligned}$$

  sisteminin denge noktaları

  $$\begin{aligned} f(x,y) &= 0 \\ g(x,y) &= 0 \end{aligned}$$

  sisteminin reel çözüm kümesidir. En genel halde

  $$dX/dt = F(X), X = (X_1, X_2, ..., X_n)^T, F = (F_1, F_2, ..., F_n)^T$$

[1] Geralamo Cardano(1501-1576) İtalyan matematikçi.

sisteminin denge noktaları

$$F(X) = 0$$

nonlineer cebirsel sisteminin reel çözüm kümesidir.

- Diferensiyel denklemler için özdeğer problemleri de sıfıryeri belirleme problemleri arasında yer alırlar. Örneğin

$$\begin{aligned} y'' + \lambda y &= 0 \\ y(0) &= 0, y(1) + y'(1) = 0 \end{aligned}$$

denkleminin sıfırdan faklı çözüme sahip olduğu ve problemin özdeğerleri olarak bilinen λ değerleri, $f(\lambda) = tan(\lambda) + \lambda$ fonksiyonunun sıfır yerleridir ve bu sıfır yerleri ancak sayısal yöntemler yardımıyla belirlenebilirler.

Bu örnekler daha da artırılabilir. Öncelikle f fonksiyonunun sıfır yeri için sabit nokta iterasyon yöntemini inceleyelim.

6.2 Sabit nokta iterasyon yöntemi

TANIM 6.1. *Bir problemin çözümünde $n+1$ inci adımda elde edilen yaklaşım, n ve/veya daha önceki adımlarda elde edilen yaklaşımları kullanıyorsa, bu tür yöntemlere iterasyon yöntemleri veya iteratif yöntemler adı verilir.*

TEOREM 6.1.

$$g : [a, b] \longrightarrow [a, b]$$

sürekli bir fonksiyon olmak üzere g nin $[a, b]$ aralığında en az bir sıfır yeri vardır.

İspat. *[Alıştırma 1].*

TANIM 6.2. *$x_0 \in [a, b]$ keyfi bir başlangıç noktası olsun.*

$$x_{n+1} = g(x_n), n = 0, 1, 2, \cdots \tag{6.1}$$

ile tanımlanan $\{x_n\}_{n=0}^{\infty}$ dizisine g fonksiyonu ve x_0 başlangıç noktası ile üretilen iterasyon dizisi adı verilir.

Önerme 6.1. *Eğer* $\{x_n\}_{n=0}^{\infty}$ *dizisi bir* r *noktasına yakınsıyorsa ve* g *fonksiyonu* r *noktasında sürekli ise bu taktirde* r *noktası* g *nin sabit noktasıdır.*

İspat.

$$r = \lim_{n\to\infty} x_{n+1} = \lim_{n\to\infty} g(x_n) = g(\lim_{n\to\infty} x_n) = g(r)$$

dir.(Yukarıdaki işlemlerde g *fonksiyonunun sürekliliğini hangi adımda kullandık?)*

Verilen bir f fonksiyonunun sıfır yerini bulmak için sıkça kullanılan yöntem, *sıfır yeri belirleme problemini sabit nokta belirleme problemine dönüştürmektir.* Böylece f nin sıfır yerini belirleme problemi uygun bir g fonksiyonu için g nin sabit noktasını belirleme problemine dönüştürülmüş olur, yani

$$f(x) = 0 \iff x = g(x)$$

dir. f fonksiyonunun sıfır yerini sabit nokta kabul eden çok sayıda g fonksiyonu bulunabilir. Ancak bu fonksiyonlardan bazıları f nin sıfır yeri için yakınsak iterasyon üretirken, diğer bir kısmı ise başlangıç noktası sıfır yerine ne kadar yakın seçilirse seçilsin ıraksak bir iterasyon üretebilir.

ÖRNEK 6.1. $f(x) = x^2 - x - 1$ *fonksiyonun sıfır yerlerini belirleme problemi-nin aşağıda verilen* g *fonksiyonlarının sabit noktalarını belirleme problemine denk olduğunu gözlemleyerek,* $x_0 = \sqrt{2}$ *ile oluşturulan iterasyonların yakınsaklığını araştırınız.*

- $g(x) = x^2 - 1$
- $g(x) = x + c(x^2 - x - 1), c \in R$ *ve*

$$c = -2/\sqrt{5}, -3/4, -1/\sqrt{5}$$

- $g(x) = \frac{x^2+1}{2x-1}$

Çözüm.

- Yukarıda verilen g fonksiyonlarının sabit noktalarının f nin sıfır yerleri olduğu kolayca görülebilir. Örneğin

$$g(x) = x^2 - 1 = x \Leftrightarrow f(x) = x^2 - 1 - x = 0$$

dır. f nin sıfır yerlerinden birisi altın oran olarak bilinen

$$r_1 = \left(1 + \sqrt{5}\right)/2 \cong 1.6180$$

ve diğeri ise ,

$$r_2 = \left(1 - \sqrt{5}\right)/2 \cong -0.6180$$

dir.

- $g(x) = x^2 - 1$ fonksiyonunu göz önüne alalım. Hiçbir x_0 ile (6.1) yardımıyla oluşturulan iterasyon $r_1 = \left(1 + \sqrt{5}\right)/2$ sabit noktasına yakınsamaz: Örneğin r_1 e yakın seçilen $x_0 = \sqrt{2}$ için

$$\begin{aligned} x_1 &= g(x_0) = g(\sqrt{2}) = 1 \\ x_2 &= g(x_1) = g(1) = 0 \\ x_3 &= g(x_2) = g(0) = -1 \\ x_4 &= g(x_3) = g(-1) = 0 \end{aligned}$$

biçimde yakınsak olmayan salınımlı bir dizi elde edilir. Daha da yakın komşulukta seçilen $x_0 = 1.6$ için de aynı salınımlı dizi elde edilir. O halde $g(x) = x^2 - 1$ iterasyon fonksiyonu f nin r_1 sabit noktasını belirlemek için uygun değildir.

- $g(x) = x + c(x^2 - x - 1)$ iterasyon fonksiyonu $x_0 = \sqrt{2}$
  - ve $c = -\frac{2}{\sqrt{5}}$ değeri için belirli bir adımdan sonra 1.6880 ve 1.5437 değerlerini alternatif olarak almak suretiyle hiç bir noktaya yakınsamaz:

$$\begin{aligned} x_1 &= g(\sqrt{2}) = 1.7847, \\ x_2 &= g(x_1) = 1.4265, \\ x_3 &= g(x_2) = 1.7767, \\ &\vdots \\ x_n &= g(x_{n-1}) = 1.6880, \\ x_{n+1} &= g(x_n) = 1.5437, \\ x_{n+2} &= g(x_{n+1}) = 1.6880, \cdots \end{aligned}$$

- Öte yandan $c = -3/4$ değeri için 57 adımda $|x_{n+1} - x_n| < \epsilon = 1E-10$ kriterini sağlayan 1.61803398878648 noktasına yakınsar.
- Son olarak, $c = -1/\sqrt{5}$ değeri için ise 5 adımda 1.61803398874989 değerine yakınsar.

O halde c parametresinin değerine göre aynı başlangıç değeriyle oluşturulan iterasyon bazen ıraksak, bazen yavaş yakınsak veya veya bazen de hızlı yakınsak bir dizi oluşturabilmektedir.

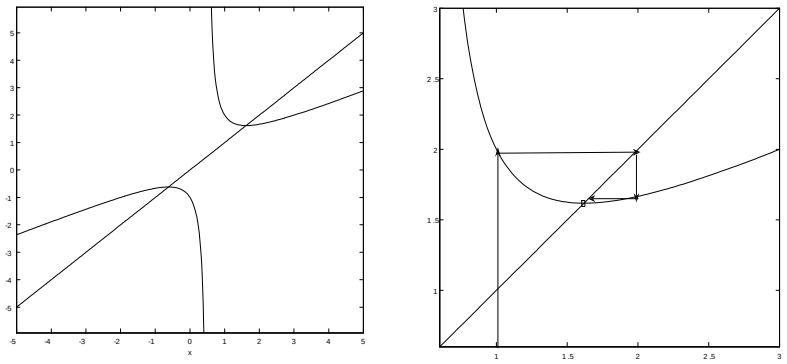
- Öte yandan

$$g(x) = \frac{x^2 + 1}{2x - 1}$$

iterasyon fonksiyonu da $x_0 = \sqrt{2}$ için 5 adımda 1.61803398874989 değerine yakınsar.

Şekil 6.1 (a) da bu g fonksiyonunu ve $h(x) = x$ ile tanımlanan h fonksiyonunun grafikleri sunulmaktadır.

- g fonksiyonunun biri pozitif ve diğeri negatif olan iki sabit noktası (g ve h fonksiyonlarının grafiklerinin kesim noktaların apsisleri) olduğu görülmektedir. Şekil 6.1(b) de $x_{n+1} = g(x_n)$ iterasyonu ile elde edilen x_n noktaları için $(x_n, g(x_n))$ çiftlerinin $(r_1, g(r_1))$ noktasına yakınsadığı görülmektedir.



Şekil 6.1: (a)g ve h fonksiyonları, (b)$(x_i, g(x_i))$ nokta çiftleri

Yukarıdaki iterasyon dizileri Program 6.1 yardımıyla elde edilmiştir.

```
%Sabit Nokta iterasyon örneği
%---------------------------------------

function x1=sabiter(x0)
 test=1;sayac=0;Max_sayac=20;
  while test
     x1=g(x0)
     fark=abs(x1-x0);sayac=sayac+1;
     test=fark> eps & sayac< Max_sayac;
     x0=x1;
  end
 function y=g(x)
    y=(x^2+1)/(2*x-1);
 end
end
%---------------------------------------
```

Program 6.1: Sabit nokta iterasyon uygulaması

Yukarıdaki örnekten, bazı g fonksiyonları ile oluşturulan iterasyonların ıraksadıkları, diğerlerinin ise farklı hızlarda yakınsadıklarını gözlemledik. Acaba yakınsak iterasyon üreten iterasyon fonksiyonları belirli kriterler yardımıyla ayırt edebilir miyiz? Hatta yakınsak olanların yakınsama hızları için ne söyleyebiliriz? Bunun için aşağıdaki Teoremi inceleyelim:

TEOREM 6.2. *[1] $g : [a, b] \to [a, b]$ sürekli ve (a, b) aralığında türevlenebilir bir fonksiyon ve $r \in [a, b]$, g nin bu araklıktaki bir sabit noktası olsun. Ayrıca $\forall x \in (a, b)$ için*

$$|g\prime(x)| \leq K < 1$$

özelliğini sağlayan K sabitinin var olduğunu kabul edelim. Bu durumda $x_0 \in [a, b]$ olmak üzere

$$x_{n+1} = g\left(x_n\right), n = 0, 1, 2, \cdots$$

ile tanımlanan iterasyon her $n \geq 0$ için

$$|x_n - r| \leq K^n |\ x_0 - r|$$

eşitsizliğini sağlar ve sonuç olarak bu iterasyonla üretilen $\{x_n\}_{n=0}^{\infty}$ dizisi r sabit noktasına yakınsar.

İspat.

*Türevler için ortalama değer teoremi*nden[12]

$$\begin{aligned} |x_1 - r| &= |g(x_0) - r| = |g(x_0) - g(r)| = |g'(c_0)||x_0 - r| \\ &\leq K|x_0 - r|, c_0 \in (a, b) \end{aligned}$$

elde ederiz. Yani, x_1 noktası r ye x_0 dan daha yakındır. Tümevarım gereği

$$|x_{n-1} - r| \leq K^{n-1}|x_0 - r|$$

olduğunu kabul edelim. Tekrar ortalama değer teoremi ve tümevarım hipotezi gereği

$$\begin{aligned} |x_n - r| &= |g(x_{n-1}) - r| = |g(x_{n-1}) - g(r)| = |g'(c_{n-1})||x_{n-1} - r| \\ &\leq K|x_{n-1} - r| \\ &\leq K^n|x_0 - r|, c_{n-1} \in (a, b) \end{aligned}$$

elde ederiz. Buradan $n \to \infty$ için limit almak suretiyle $0 < K < 1$ olduğundan $\lim_{n\to\infty} |x_n - r| = 0$ elde ederiz ve *sıkıştırma teoremi*[12] yardımıyla $\lim_{n\to\infty} x_n = r$ olduğu görülür.

İterasyonun yakınsaması için bir diğer yeter şart aşağıdaki teoremle ifade edilmektedir:

Sonuç 6.1. *$g : [a, b] \to [a, b]$ sürekli ve (a, b) aralığında türevlenebilir ve türevi sürekli bir fonksiyon ve $r \in (a, b)$, g nin bu araklıktaki bir sabit noktası ve ayrıca $|g\prime(r)| < 1$ olsun. Bu durumda r ye yeterince yakın $x_0 \in [a, b]$ için*

$$x_{n+1} = g(x_n), n = 0, 1, 2, \cdots$$

ile üretilen $\{x_n\}_{n=0}^{\infty}$ dizisi r sabit noktasına yakınsar.

İspat. *$|g\prime(r)| < 1$ ve g' fonksiyonu $[a, b]$ de sürekli olduğundan*

$$|g\prime(x)| \leq K < 1, \forall x \in (r - \delta, r + \delta)$$

sağlanacak biçimde $\delta > 0$ sabiti mevcuttur. $x_0 \in (r - \delta, r + \delta)$ ile Teorem 6.2 uygulanarak sonuç elde edilir.

Uyarı. *$|g'(r)| = 1$ olması durumunda üretilen iterasyon başlangıç nokta seçimine göre yakınsak veya ıraksak olabilir(Alıştırma 7).*

Sonuç 6.2. *$g : [a, b] \to [a, b]$ sürekli ve (a, b) aralığında türevlenebilir ve türevi sürekli bir fonksiyon ve $r \in [a, b]$, g nin bu araklıktaki bir sabit noktası ve $|g\prime(r)| > 1$ olsun. Bu durumda $x_0 \in [a, b]$, $x_0 \neq r$ için*

$$x_{n+1} = g(x_n), n = 0, 1, 2, \cdots$$

ile üretilen $\{x_n\}_{n=0}^{\infty}$ dizisi r sabit noktasına yakınsamaz.

İspat. *Alıştırma 8.*

ÖRNEK 6.2. *Teorem 6.2 yardımıyla Örnek 6.1 e ait sonuçları analiz ediniz.*

Çözüm.

- $g(x) = x^2 - 1$ fonksiyonu ile üretilen dizi ıraksadı çünkü

  $$g'((1 + \sqrt{5})/2) = 1 + \sqrt{5} > 1$$

  dir ve yukarıdaki sonuca göre hiç bir x_0 ile üretilen dizi yakınsamaz. Aynı sonuç diğer sabit nokta için de geçerlidir, çünkü

  $$|g'((1 - \sqrt{5})/2)| = |1 - \sqrt{5}| > 1$$

  dir.

- İlk c değeri, yani $c = -\frac{2}{\sqrt{5}}$ için

  $$g(x) = x - \frac{2}{\sqrt{5}}(x^2 - x - 1)$$

  olup,

  $$g'(x) = \frac{-4}{\sqrt{5}}x + \frac{2}{\sqrt{5}} + 1$$

  ve

  $$g'((1 + \sqrt{5})/2) = -1$$

  elde edilir. İterasyon bu durumda ıraksamıştır. Aynı sonuç diğer sabit nokta için de geçerlidir, çünkü

  $$g'((1 - \sqrt{5})/2) = 3 > 1$$

  dir.

– Öte yandan $c = -\frac{3}{4}$ için

$$g(x) = x - \frac{3}{4}(x^2 - x - 1)$$

olup,

$$g'(x) = -\frac{3}{2}x + \frac{7}{4}$$

ve

$$|g'((1+\sqrt{5})/2)| = |-0.677| < 1$$

elde edilir ve bu durumda iterasyonun pozitif sabit noktası için yakınsamış olması beklenen bir sonuçtur. Ancak

$$g'((1-\sqrt{5})/2) \doteq 2.677 > 1$$

dir ve iterasyon negatif sabit nokta için yakınsamaz.

– Fakat $c = -\frac{1}{\sqrt{5}}$ için

$$g(x) = x - \frac{1}{\sqrt{5}}(x^2 - x - 1)$$

olup,

$$g'(x) = \frac{-2}{\sqrt{5}}x + \frac{1}{\sqrt{5}} + 1$$

ve

$$g'((1+\sqrt{5})/2) = 0$$

elde edilir ve bu durumda da iterasyonun yakınsamış olması sürpriz değildir. Ancak $-0.618 < x_0 \leq 3.8$ için iterasyon $(1+\sqrt{5})/2$ ye yakınsarken, $x_0 < -0.619$ ve $x_0 > 3.9$ için iterasyon ıraksamaktadır. Öte yandan bu g fonksiyonu sadece pozitif sabit noktasını belirlemek için kullanılabilmektedir, çünkü

$$g'((1-\sqrt{5})/2) = 2 > 1$$

dir.

- Öte yandan

$$g(x) = \frac{x^2+1}{2x-1}$$

iterasyon fonksiyonu için,

$$g'(x) = \frac{(x-1)^2 + x^2 - 1}{(2x-1)^2}$$

olup,

$$g'((1+\sqrt{5})/2) = 0$$

dır. Ayrıca $(1, \infty)$ aralığındaki her x için $0 < g'(x) < 1$ dir.Dolayısıyla herhangi $x_0 \in (1, \infty)$ için iterasyon yakınsaktır. $x_0 = 0.6$ için $|g'(0.6)| = 12 > 1$ olmasına rağmen iterasyon bu başlangıç noktası ile de yakınsaktır. Ayrıca aynı iterasyon fonksiyonu negatif sabit nokta için de yakınsak iterasyon üretir, çünkü

$$g'((1-\sqrt{5})/2) = 0 < 1$$

dir.

- İterasyon fonksiyonu ile üretilen dizinin, fonksiyonun sahip olduğu her bir sabit nokta için yakınsak iterasyon üretmesi beklenmemelidir.
- Yakınsak iterasyon genelde sadece sabit noktanın yeterince yakın komşuluğunda seçilen x_0 başlangıç noktaları için elde edilebilir.
- r noktası g nin sabit noktası olmak üzere $|g'(r)| < 1$ şartını sağlayan g iterasyon fonksiyonu ile yakınsak dizi üretebilmek için r nin yeterince yakın komşuluğunda seçilen x_0 için $|g'(x_0)| < 1$ şartı, genelde yeterlidir ancak gerekli değildir.

6.3 İterasyon fonksiyonu seçimi ve *Newton-Raphson* yöntemi

Yukarıdaki örneklerden, bazı iterasyon fonksiyonlarının başlangıç noktası sabit noktaya çok yakın seçilmesine rağmen, sabit noktaya yakınsayan iterasyon üretmeyebileceğini gözlemledik. Öte yandan, bir iterasyon fonksiyonunun bir sabit nokta için yakınsak iterasyon üretirken, bir diğer sabit nokta için ıraksak bir iterasyon oluşturabileceğine dikkat ettik. Ayrıca yakınsak iterasyon üretmesine rağmen, farklı iterasyon fonksiyonlarının farklı hızlarda yakınsayan diziler oluşturduğunu da gözlemledik. Ancak bunları incelerken iterasyon fonksiyonlarının nasıl seçildiği üzerinde durmadık.

Bu bağlamda *Newton*[2], hala güncelliğini koruyan bir öneri sunmuştur: Sıfır noktası komşuluğunda türevlenebilen f fonksiyonunun sıfır yerini bulma problemi

$$g(x) = x - f(x)/f'(x)$$

ile tanımlanan g fonksiyonunun sabit noktasını bulma problemine denktir, diğer bir değimle her iki problem de aynı çözüm kümesine sahiptirler. Bu özel g iterasyon fonksiyonu ve sıfıryerine yeterince yakın seçilen x_0 başlangıç noktası ile oluşturulan

$$x_{n+1} = x_n - f(x_n)/f'(x_n), n = 0, 1, 2, ...$$

iterasyonuna *Newton* veya *Newton-Raphson*[3] iterasyonu ve g fonksiyonuna da *Newton-Raphson* iterasyon fonksiyonu adı verilmektredir.

Yukarıdaki örneğimizde son seçenek olarak kullandığımız

$$g(x) = \frac{x^2 + 1}{2x - 1} = x - \frac{x^2 - x - 1}{2x - 1}$$

fonksiyonu esasen *Newton-Raphson* iterasyon fonksiyonudur, dolayısıyla bu fonksiyon seçimiyle yukarıda *Newton-Raphson* yöntemini uygulamış olduk ve iterasyonun her iki sabit noktaya da hızlı bir biçimde yakınsadığını gözlemledik.

Bu yönteme göre sıfır yerine yeterince yakın seçilen x_0 için elde edilen

$$x_1 = x_0 - f(x_0)/f'(x_0)$$

yaklaşımı Şekil 6.2 de de görüldüğü üzere $(x_0, f(x_0))$ noktasında $y = f(x)$ fonksiyonun grafiğine çizilen teğet doğrusunun(yani,

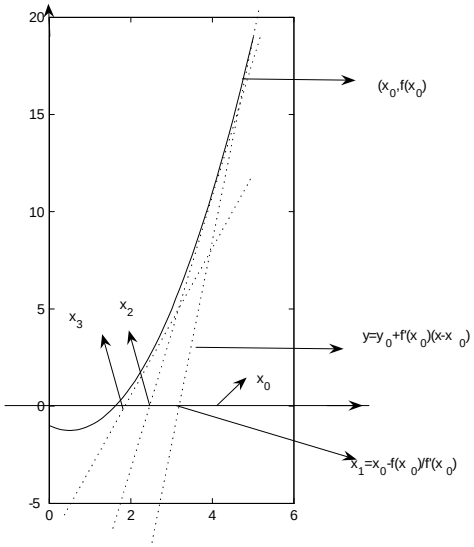
$$y = f(x_0) + f'(x_0)(x - x_0)$$

ile tanımlanan doğrunun) x eksenini kesim noktasıdır.

Diğer bir bakış açısıyla f fonksiyonunun sıfıryerini doğrudan bulmak yerile, x_0 noktasındaki Taylor açılımının lineer kısmının sıfıryeri x_1 yaklaşımı olarak kabul edilir. Daha sonra x_1 noktasındaki Taylor açılımının lineer kısmının sıfıryeri x_2 olarak kabul edilir ve dizinin diğer elemanları benzer biçimde elde edilir, Şekil 6.2 .

[2]Isaac Newton 1642 – 1726, İngiliz fizikçi ve matematikçi

[3]Joseph Raphson, 1648-1715, İngiliz matematikçi.



Şekil 6.2: Newton-Raphson yöntemi ile f nin sıfır yeri için yaklaşımlar

Benzer biçimde yöntem x_2 yaklaşımını, $(x_1, f(x_1))$ noktasından çizilen teğetin x eksenini kesim noktası olarak kabul eder. Bu yaklaşımlar eğer bir noktaya yakınsıyorlarsa, yakınsadıkları nokta g fonksiyonunun sabit noktası ve dolayısıyla da f nin sıfıryeridir.

- r noktası f nin sıfıryeri ve $f'(r) \neq 0$, $f''(r)$ tanımlı ise $g'(r) = 0$ dır.

$$g(x) = x - f(x)/f'(x)$$

olup, $f(r) = 0$ olduğundan

$$g'(r) = 1 - \Big((f'(r))^2 - f(r)f''(r)\Big)\,(f'(r))^2 = 0$$

olarak elde edilir.

- *Newton-Raphson* yöntemi sıfır noktasının küçük komşuluğunda kuadratik(ikinci basamaktan) yakınsak bir yöntemdir: r noktası g fonksiyonunun sabit noktası, $f'(r) \neq 0$ ve x_n ise $n - inci$ adımda elde edilen yaklaşım olmak üzere

$$\begin{aligned} x_{n+1} &= g(x_n) \\ &= g(r) + g'(r)(x_n - r) + \frac{1}{2}g''(c_{x_n})(x_n - r)^2, \end{aligned}$$

burada c_{x_n} , x_n ve r arasında bir noktadır. $g(r) = r$ ve $g'(r) = 0$ olduğunu kullanarak

$$x_{n+1} - r = \frac{1}{2}g''(c_x)(x_n - r)^2$$

elde ederiz. Dolayısıyla $n + 1$ -inci adımdaki hata $n-$inci adımdaki hatanın karesiyle orantılıdır. Bu sonuç ise yöntemin, eğer yakınsak ise, kuadratik olarak yakınsadığını gösterir.

- *Newton-Raphson* iterasyonunun r noktasına yakınsaması için yeter şart x_0 noktasının r noktasını içeren ve

  $$|g'(x)| = |f(x)f''(x)|/(f'(x))^2 < 1$$

  şartını sağlayan komşulukta seçilmesidir. Bu şart yukarıda da belirtildiği gibi gerekli değildir: Örnek 6.2 de $x_0 = 0.6$ için $|g'(x_0)| > 1$ dir, ancak iterasyon yine de yakınsaktır.

- Eğer bir g iterasyon fonksiyonu r sabit noktasının komşuluğunda $n-inci$ mertebeden türeve sahip ve

  $$g'(r) = g''(r) = \cdots = g^{(n-1)}(r) = 0, g^{(n)}(r) \neq 0$$

  ise g ile elde edilen iterasyon $n-inci$ basamaktan yakınsaktır(Alıştırma 9). $n = 1$ olması durumunda yöntem *lineer yakınsaktır* denir. Bu sonuç yukarıda *Newton* yöntemi için elde edilen sonucun genel halidir ve benzer biçimde ispatlanabilir.

- *Newton-Raphson* fonksiyonundan farklı olarak

  $$g(x) = x - \frac{1}{\sqrt{5}}(x^2 - x - 1)$$

  ile pozitif sabit nokta için üretilen iterasyon da ikinci basamaktan yakınsaktır, çünkü $g'((1+\sqrt{5})/2) = 0$ ve $g''((1+\sqrt{5})/2) = -2/\sqrt{5} \neq 0$ dır.

ÖRNEK 6.3. *$r > 0$ olmak üzere*

$$f(r) = 10000 - \frac{350}{r}\left(1 - 1/(1+r)^{36}\right)$$

fonksiyonu verilmiş olsun. Bu tür fonksiyonlara matematiksel finans işlemlerinde sıkça karşılaşılır. f nin $r_0 = 0.1$ başlangıç noktası ile sıfır yerini bulunuz.

Çözüm.

Öncelikle

$$f\prime(r) = 350\frac{1 - \frac{1}{(1+r)^{36}}}{r^2} - \frac{12600}{r(1+r)^{37}}$$

olarak elde edildiğine dikkat edelim. $r_0 = 0.1$ başlangıç değeri için

$$r_{i+1} = r_i - f(r_i)/f'(r_i), i = 0, 1, 2, ...$$

ile tanımlanan *Newton-Raphson* yaklaşımları olarak elde edilir.

i	r_i
0	0.1
1	−0.1193
2	−0.0900
3	−0.0594
4	−0.0293
5	−0.0040
...	...
8	0.0131
9	0.0131

Hatırlatma 6.1. *Genel olarak iterasyon yöntemleri karmaşık sıfır yerlerini belirlemek için de kullanılabilirler. Bu sonuç özel olarak Newton-Raphson yöntemi için de geçerlidir, ancak bunun için başlangıç noktasının da karmaşık bir sayı olarak seçilmesi gerekir.*

ÖRNEK 6.4. *$f(x) = x^2 + 1$ fonksiyonunun $x_0 = 1 + i$ noktası komşuluğundaki sıfıryerini belirleyiniz.*

Çözüm.

Newton yöntemi yardımıyla aşağıdaki yaklaşımları elde ederiz:

i	x_i
0	$1+i$
1	$0.2500+0.7500i$
2	$-0.0750+0.9750i$
3	$0.0017+0.9973i$
4	$-0.0000+1.0000i$

6.3.1 Katlı kökler için *Newton-Raphson* yöntemi

Mevcut haliyle *Newton-Raphson* yöntemi katlı kökler için lineer olarak yakınsayan bir yöntemdir. Eğer r noktası f fonksiyonunun $m>1$ katlı bir sıfıryeri ise

$$f(x)=(x-r)^m h(x), h(r)\neq 0$$

olacak biçimde h fonksiyonu vardır. Bu durumda g iterasyon fonksiyonu

$$\begin{aligned} g(x) &= x-f(x)/f'(x) \\ &= x-\frac{(x-r)^m h(x)}{m(x-r)^{m-1}h(x)+h'(x)(x-r)^m} \\ &= x-\frac{(x-r)h(x)}{mh(x)+h'(x)(x-r)} \end{aligned}$$

olarak yazılabilir ve

$$\begin{aligned} g'(x) &= 1-\left[\begin{array}{c}(h(x)+(x-r)h'(x))(mh(x)+h'(x)(x-r)) \\ -(mh'(x)+h''(x)(x-r)+h'(x))(x-r)h(x)\end{array}\right] \\ &\quad /(mh(x)+h'(x)(x-r))^2 \end{aligned}$$

elde edilir. Buradan

$$\begin{aligned} g'(r) &= 1-h(r)mh(r)/(mh(r))^2 \\ &= 1-1/m\neq 0 \end{aligned}$$

olur. $g'(r)<1$ olduğu için *Newton-Raphson* iterasyonu katlı kökler uygun başlangıç noktası ile sadece lineer olarak yakınsaktır. Bu durumda yukarıda tanımlanan g iterasyon fonksiyonu yeniden düzenlenerek

$$g(x)=x-mf(x)/f'(x) \tag{6.2}$$

olarak yazılırsa, yine $g(r) = r$ olduğu açıktır. Ayrıca yukarıdaki işlemler tekrar edilerek $g'(r) = 0$ elde ederiz. Bu sonuç ise yöntemin uygun başlangıç noktası ile en az kuadratik olarak yakınsayacağını ifade eder. Bir örnek üzerinde her iki yöntemin performansını karşılaştırmak mümkündür.

ÖRNEK 6.5. *$f(x) = x^2 - 2x + 1$ fonksiyonu için $x_0 = 2$ alarak basit kökler ve katlı kökler için Newton-Raphson yöntemiyle sıfıryeri yaklaşımlarını hesaplayınız.*

Çözüm.

Basit kökler için kuadratik olarak yakınsak olan *Newton-Raphson* yöntemini $x = 1$ noktasında *iki katlı köke sahip f fonksiyonu* için uygulayarak aşağıdaki tabloda verilen ve son sütundan görüldüğü üzere **lineer olarak yakınsayan** yaklaşımları elde ederiz, yakınsama lineerdir çünkü $(x_{n+1} - r)/(x_n - r)$ oranı yaklaşık olarak sabittir:

İterasyon	$Newton(BasitKökler)$	$(x_{n+1} - r)/(x_n - r), r = 1$
0	2.0000	
1	1.5000	$0.5/1 = 0.5$
2	1.2500	$0.25/0.5 = 0.5$
3	1.1250	$0.125/0.250 = 0.5$
4	1.0625	$0.625/0.1250 = 0.5$
5	1.0313	$0.313/0.625 = 0.5008$
6	1.0156	$0.0156/0.0313 = 0.4984$
7	1.0078	$0.0078/0.0156 = 0.5$
8	1.0039	$0.0039/0.0078 = 0.5$
9	1.0020	$0.0020/0.0039 = 0.5128$
10	1.0010	$0.0010/0.0020 = 0.5$

Tablo 6.1: Basit kökler için Newton yaklaşımları

Öte yandan aynı başlangıç değeri ile $m = 2$ değeri için yukarıda verilen $x_0 = 2$ başlangıç noktası ile uygulayarak ilk adımda $x_1 = 1$ gerçek sıfıryerini elde ederiz.

Katlı sıfır yerleri belirlemek için basit sıfır yeri için geliştirilen versiyonun kullanılması uygun olmadığı gibi basit sıfır yerini belirlemek için de, katlı sıfır yeri için (6.2) ile düzenlenen versiyon kullanılırsa istenilen sonuçlar elde edilemeyebilir.

Örneğin $f(x) = x^2 - x - 1$ fonksiyonu ve $x_0 = \sqrt{2}$ başlangıç noktası ve basit sıfıryerleri için *Newton-Raphson* yöntemi Örnek 6.1 de belirtildiği gibi 5

adımda yakınsarken, $m = 2$ ile katlı sıfyerleri için düzenlenen versiyon 2.0000 ve 1.3333 değerleri arasında alternatif olarak salınım yapmaktadır(Alıştırma 14).

6.3.2 Sayısal türev ile yaklaşım(Kiriş yöntemi)

Newton-Raphson yönteminde $f'(x_i)$ türev değeri şuphesiz fonksiyonun türevinin de bilinmesini gerektirmektedir. Türev bilgisi gerektirmeyen alternatif bir yöntem ise

$$f'(x_i) \cong \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}}$$

yaklaşımının kullanılmasıdır. Bu durumda tek birbaşlangıç değeri yerine x_0 ve x_1 başlangıç değerleri ile başlatılabilen

$$x_{i+1} = x_i - \frac{f(x_i)}{\frac{f(x_i)-f(x_{i-1})}{x_i - x_{i-1}}} = x_i - \frac{x_i - x_{i-1}}{f(x_i) - f(x_{i-1})} f(x_i), i = 2, 3, \cdots \qquad (6.3)$$

yaklaşımları elde edilir. Bu yöntem 1. Bölümde de değindiğimiz üzere *kiriş yöntemi* olarak bilinir. *Newton-Raphson* yönteminde olduğu gibi $(x_i, f(x_i))$ noktasındaki teğetin x eksenini kesim noktası yerine, $(x_{i-1}, f(x_{i-1})), (x_i, f(x_i))$ noktalarından geçen ve

$$y - f(x_i) = \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}}(x - x_i)$$

ile verilen kiriş doğrusunun x eksenini kesim noktası sıfır yeri için x_{i+1} yaklaşımı olarak kabul edilir.

ÖRNEK 6.6. *$f(x) = x^2 - x - 1$ fonksiyonu için $x_0 = 3, x_1 = 2$ başlangıç tahminleri ve kiriş yöntemi ile x_2, x_3, x_4 ve x_5 sıfır yeri yaklaşımlarını hesaplayınız.*

Çözüm.

İlk yaklaşım

$$\begin{aligned} x_2 &= x_1 - \frac{x_1 - x_0}{f(x_1) - f(x_0)} f(x_1) \\ &= 2 - \frac{2-3}{f(2) - f(3)} f(2) \\ &= 2 - \frac{-1}{1-5} = 2 - \frac{1}{4} = \frac{7}{4} \cong 1.75 \end{aligned}$$

olarak elde edilir. Benzer biçimde diğer yaklaşımlar

$$x_3 = 5/3 \cong 1.6667,\ x_4 = 47/29 \cong 1.6207,\ x_5 = 322/199 \cong 1.6181.$$

olarak elde edilir.

6.3.3 Başlangıç noktası seçimi

Newton-Raphson yönteminin en zayıf yönü, başlangıç noktası seçiminin sıfıryerine "yakın" seçilmesini gerektirmesidir. Tahmini noktanın ne kadar "yakın" seçilmesi gerektiği problemden probleme göre değişmektedir. Örneğin altın oranı sıfıryeri kabul eden örneğimizde(Örnek 6.1), $x_0 \in (1, \infty)$ olması durumunda altın orana yakınsayabildiğimizi görmüştük. Dolayısıyla bu problem için başlangıç noktası seçimi herhangi bir sorun teşkil etmemiştir.

Ancak bazı fonksiyonlar başlangıç noktası tahmininin iyi yapılmış olmasını gerektirirler.

ÖRNEK 6.7. $f(x) = ((x-2)^2 + 1/100)(x-3)$ *fonksiyonu için farklı başlangıç noktası seçimi ile Newton-Raphson yönteminin yakınsaklığını araştırınız.*

Çözüm.

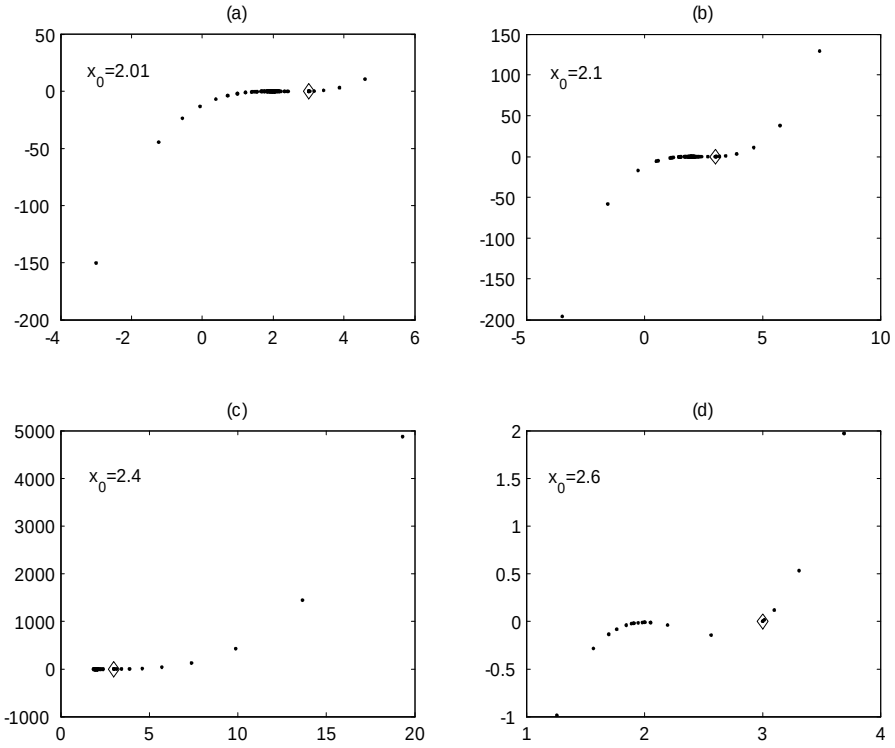
Fonksiyonunun $x = 3$ noktasındaki sıfır yerini $|f(x_n)| < 10^{-6}$ kriterini sağlayacak yaklaşımlar elde etmek için gerekli iterasyon sayıları(iter) aşağıdaki tabloda verilmektedir.

x_0	2	2.01	2.02	2.04	2.06	2.08	2.1	2.2	2.3	2.4	2.5	2.9
iter	1	144	15	27	9	52	129	12	28	44	10	4

Yukarıdaki tablodan sıfır yeri komşuluğunda seçilen farklı noktalar ile *Newton-Raphson* iterasyonunun $|f(x_n)| < 10^{-6}$ kriteri ile yakınsaması için farklı sayıda iterasyon sayıları gerekmektedir. Ancak gerekli iterasyon sayılarının düzensiz bir dağılıma sahip olduğu görülmektedir.

Örneğin

- $x_0 = 2$ için sadece 1 adet iterasyon gerekirken,
- $x_0 = 2.01$ için 144 adet ve
- $x_0 = 2.02$ için ise 15 adet iterasyon gerekmektedir.



Şekil 6.3: Farklı başlangıç değerleri ile x_i Newton-Raphson yaklaşımları için $(x_i, f(x_i))$ değerleri:(a) $x_0 = 2.01$,(b) $x_0 = 2.1$ (c) $x_0 = 2.4$ (d) $x_0 = 2.6$

Ayrıca

- Herbir başlangıç noktası için elde edilen iterasyonlar yakınsak, ancak yakınsama hızları çok farklıdır.
- Yine yakınsamanın monoton değil, salınımlı olduğu görülmektedir.

Gözlem 6.1. *Sonuç olarak örneğimizde başlangıç noktası seçiminin iterasyonunu yakınsama hızı üzerinde etkisinin çok fazla olduğu görülmektedir. Bunun nedeni sıfır noktasının geniş bir komşuluğunda f fonksiyonun sıfıra çok yakın değerler alması ve f' türevinin $x_1 = 2.005, x_2 = 2.6616$ sıfır yerlerine sahip olması ve dolayısıyla $f(x)/f'(x)$ fonksiyonunun süreksizliklere sahip olmasıdır.*

6.4 Nonlineer sistemler için *Newton* yöntemi

İki değişkenli f ve g fonksiyonları için

$$\begin{aligned} f(x,y) &= 0 \\ g(x,y) &= 0 \end{aligned}$$

nonlineer sisteminin (p,q) ile gösterilen sıfıryerine sahip olduğunu kabul edelim.

Örneğin

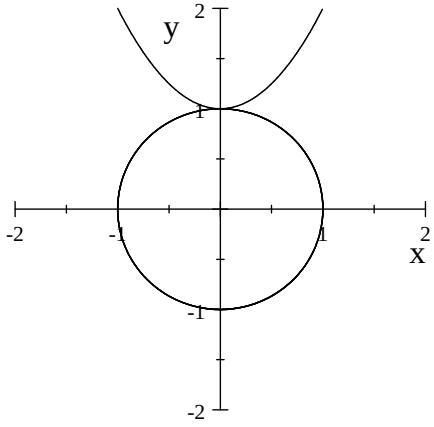
$$\begin{aligned} x^2+y^2-1 &= 0 \\ -x^2+y-1 &= 0 \end{aligned}$$

nonlineer sisteminin tek çözümü Şekil 6.4 ile belirtilen çember ve parabolün tek bir kesişim noktası olan $(0,1)$ noktasıdır.

Ayrıca bu noktanın bir D komşuluğunda

$$J = \begin{bmatrix} f_x & f_y \\ g_x & g_y \end{bmatrix}$$

Jacobien matrisinin tersinir olduğunu kabul edelim. Tek değişkenli fonksiyonlarda olduğu gibi, bu defa da nonlineer sistemin (x_0, y_0) noktasındaki Taylor açılımının lineer kısmının sıfır yerini (x_1, y_1) yaklaşımı olarak alalım:



Şekil 6.4: Birim çember ve parabol

$$\begin{aligned} f(x,y) &= f(x_0,y_0) + (x-x_0)f_x(x_0,y_0) + (y-y_0)f_y(x_0,y_0) + \cdots \quad (6.4) \\ g(x,y) &= g(x_0,y_0) + (x-x_0)g_x(x_0,y_0) + (y-y_0)g_y(x_0,y_0) + \cdots \end{aligned}$$

Taylor açılımının lineer kısımları ile oluşturulan

$$\begin{aligned} f(x_0,y_0) + (x-x_0)f_x(x_0,y_0) + (y-y_0)f_y(x_0,y_0) &= 0 \qquad (6.5) \\ g(x_0,y_0) + (x-x_0)g_x(x_0,y_0) + (y-y_0)g_y(x_0,y_0) &= 0 \end{aligned}$$

lineer sistemini gözönüne alalım ve bu sistemin (x,y) çözümünü (x_1,y_1) ile gösterelim.

$$X = \begin{bmatrix} x \\ y \end{bmatrix}, \Delta X = \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} x-x_0 \\ y-y_0 \end{bmatrix}, F(x,y) = \begin{bmatrix} f(x,y) \\ g(x,y) \end{bmatrix}$$

notasyonu ile (6.5) sistemi,

$$J(X^{(0)})\Delta X = -F(X^{(0)}) \qquad (6.6)$$

olarak yazılabilir. Bu sistem çözülerek elde edilen ΔX ile

$$X^{(1)} = X^{(0)} + \Delta X$$

yaklaşımı veya eleman bazında yazılarak

$$X^{(1)} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

elde edilir. Benzer biçimde diğer yaklaşımlar, $n = 0, 1, \cdots$ için

$$J(X^{(n)})\Delta X = -F(X^{(n)}) \tag{6.7}$$

$$X^{(n+1)} = X^{(n)} + \Delta X$$

olarak elde edilir.

ÖRNEK 6.8.

$$\begin{aligned} x^2 + y^2 &= 1 \\ x - y &= 0 \end{aligned}$$

denklem sisteminin çözümü için $(x_1, y_1), (x_2, y_2)$ *Newton yaklaşımlarını* $(x_0, y_0) = (1, 2)$ *başlangıç noktası için hesaplayınız.*

Çözüm.

Verilen sistemi

$$\begin{aligned} f(x, y) &= 0 \\ g(x, y) &= 0 \end{aligned}$$

olarak yazarsak,

$$F(x, y) = \begin{bmatrix} f(x, y) \\ g(x, y) \end{bmatrix} = \begin{bmatrix} x^2 + y^2 - 1 \\ x - y \end{bmatrix}$$

ile tanımladığımız vektör değerli fonksiyon için

$$F(x_0, y_0) = F(1, 2) = \begin{bmatrix} f(1, 2) \\ g(1, 2) \end{bmatrix} = \begin{bmatrix} 1^2 + 2^2 - 1 \\ 1 - 2 \end{bmatrix} = \begin{bmatrix} 4 \\ -1 \end{bmatrix}$$

elde ederiz. Ayrıca

$$J = \begin{bmatrix} f_x & f_y \\ g_x & g_y \end{bmatrix} = \begin{bmatrix} 2x & 2y \\ 1 & -1 \end{bmatrix}$$

olup,

$$J(x_0, y_0) = J(1, 2) = \begin{bmatrix} 2 & 4 \\ 1 & -1 \end{bmatrix}$$

olur. Bu durumda

$$J(X^{(0)})\Delta X = -F(X^{(0)})$$

denklem sistemi

$$\begin{bmatrix} 2 & 4 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = - \begin{bmatrix} 4 \\ -1 \end{bmatrix}$$

sistemine dönüşür. Bu lineer sistemi uygun bir yöntemle çözerek $\Delta x = 0$, $\Delta y = -1$ elde ederiz.O halde

$$X^{(1)} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

elde ederiz. Benzer biçimde

$$F(X^{(1)}) = F(x_1, y_1) = F(1, 1) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

ve

$$J(x_1, y_1) = J(1, 1) = \begin{bmatrix} 2 & 2 \\ 1 & -1 \end{bmatrix}$$

olup,

$$J(X^{(1)})\Delta X = -F(X^{(1)})$$

denklem sistemi

$$\begin{bmatrix} 2 & 2 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = - \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

sistemine dönüşür. Bu sistem de çözülerek, $\Delta x = -1/4$, $\Delta y = -1/4$ olarak elde edilir. O halde

$$X^{(2)} = \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \begin{bmatrix} -1/4 \\ -1/4 \end{bmatrix} = \begin{bmatrix} 3/4 \\ 3/4 \end{bmatrix}$$

elde ederiz.

Yukardaki nonlineer sistemin çözümü için gerekli yaklaşmları *Newton* yöntemiyle ve aşağıdaki *Newtons.m* isimli program yardımıyla elde edilebilir.

MATLAB/Octave ortamında belirleyen ve grafik ortamında gösteren Program 6.2 aşağıda verilmektedir.

```
% Nonlineer sistemler için Newton yöntemi uygulaması
%-------------------------------------------------

function X0=newtons(x,y)
X0=[x y]';test=1;eps=1e-5;Max_say=15;sayac=0;
plot(X0(1),X0(2),'*g');hold on;
while test
     DelX=-J(X0)\F(X0)';
     X1=X0+DelX;
     plot(X1(1),X1(2),'og');hold on;
     fark=norm(X1-X0);
     sayac=sayac+1;
     test=(fark>eps)&(sayac < Max_say);
    X0=X1
end
function Z=F(X)
 x=X(1);y=X(2);
 Z(1)=x^2+y^2-1;
 Z(2)=x-y;
function Z=J(X)
 x=X(1);y=X(2);
 Z=[2*x  2*y; 1 -1];
```

Program 6.2: Nonlineer sistemler için Newton yöntemi uygulaması

Alıştırmalar 6.1.

1. *$g : [a,b] \twoheadrightarrow [a,b]$ sürekli bir fonksiyon ise g nin $[a,b]$ aralığında en az bir sabit noktası olduğunu gösteriniz.(İpucu, $g(a) = a$ veya $g(b) = b$ ise bu durumda en az bir sabit nokta zaten mevcuttur. $g(a) > a$ ve $g(b) < b$ olduğunu kabul ederek $f(x) = x - g(x)$ fonksiyonunun $[a,b]$ de en az bir sıfıryeri olması gerektiğini aradeğer teoremi yardımıyla gösteriniz.)*

2. *Aşağıda verilen g fonksiyonlarının aynı sabit noktalara sahip olduklarını gösteriniz.*

   (a) $g(x) = -\frac{1}{5}x(x-3)(x+3)$
   (b) $g(x) = -\frac{1}{8}x(x^2-12)$
   (c) $g(x) = -\frac{1}{5}x(2x^2-13)$

3. *Soru 2 de verilen her bir g fonksiyonu için g nin sabit noktalarını sıfır yeri kabul eden bir f fonksiyonunu belirleyiniz.*

4. *Soru 3 de belirlediğiniz f fonksiyonunun sıfır yerlerini bulmak için hangi g leri kullanabilirsiniz? Hangisini kullanmayı tercih edersiniz?*

5. *$f(x) = x^2 - 5$ fonksiyonunun pozitif sıfıryerini bulmak için $g(x) = x + cf(x)$ iterasyon fonksiyonu ve seçilen c için $|g\prime(x_0)| < 1$ şartını sağlayan başlangıç noktası ile oluşturulan $x_{n+1} = g(x_n)$ iterasyonu gözönüne alalım.*

   - *İterasyonun yakınsaması için c hangi aralıkta değerler almalıdır?*
   - *Kuadratik olarak yakınsak olan iterasyon üretebilmek için c ne olmalıdır?*

6. *Soru 5 de verilen f fonksiyonunun sıfıryerlerinin aşağıda verilen g fonksiyonlarının sabit noktaları olduğunu gösteriniz. Hangi g ler ile oluşturulan iterasyonlar yakınsar? Yakınsak iterasyonlar için yakınsama basamaklarını belirleyiniz.*

   - $g(x) = x(6 - x^2)$
   - $g(x) = \frac{1}{8}x(13 - x^2)$
   - $g(x) = \frac{1}{2}x(3 - \frac{1}{5}x^2)$

7. $g(x) = x - (x-1)^2$ *iterasyon fonksiyonu verilsin.*

   - *Verilen iterasyon fonksiyonunun* $r = 1$ *sabit noktasına sahip olduğunu ve* $g'(r) = 1$ *olduğunu gözlemleyiniz.*
   - $r = 1$ *noktasına yeterince yakın ve* $x_0 > 1$ *başlangıç noktası için oluşturulan iterasyonun* $r = 1$ *sabit noktasına yakınsadığını uygun bir* $x_0 > 1$ *ile* x_1, x_2, x_3 *değerlerini hesaplayarak gözlemleyiniz.*
   - $x_0 < 1$ *şartını sağlayan hiçbir başlangıç noktası için* g *ile oluşturulan iterasyonun* $r = 1$*sabit noktasına yakınsamadığını uygun bir* $x_0 > 1$ *ile* x_1, x_2, x_3 *değerlerini hesaplayarak gözlemleyiniz.*

8. $g : [a, b] \to [a, b]$ *sürekli ve* (a, b) *aralığında türevlenebilir ve türevi sürekli bir fonksiyon ve* $r \in [a, b]$, g *nin bu araklıktaki bir sabit noktası olsun. Ayrıca* $|g\prime(r)| > 1$ *olduğunu kabul edelim. Bu durumda* $x_0 \in [a, b]$, $x_0 \neq r$ *için* $x_{n+1} = g(x_n), n = 0, 1, 2, \cdots$ *ile üretilen* $\{x_n\}$ *dizisi* r *sabit noktasına yakınsamayacağını gösteriniz.*

9. *Eğer bir* g *iterasyon fonksiyonu* r *sabit noktasının komşuluğunda* $n - inci$ *mertebeden türeve sahip ve*

$$g'(r) = g''(r) = \cdots = g^{(n-1)}(r) = 0, g^{(n)}(r) \neq 0$$

   ise g *ile elde edilen iterasyon* $n - inci$ *basamaktan yakınsak olduğunu gösteriniz.*

10. $g(x) = x - 1/2\sin(2x)$ *iterasyon fonksiyonu ile oluşturulan iterasyonların* $r = 0$ *ve* $r = \pi$ *sabit noktalarına yakınsak, ancak* $r = \pi/2$ *sabit noktasına yakınsak olmadığını ilgili teori yardımıyla gösteriniz.*

11. $g(x) = x - tan(2x)$ *iterasyon fonksiyonu verilsin.*

    - $r = 0$ *ın* g *nin bir sabit noktası olduğunu ve* $g'(0) = -1$ *olduğunu gözlemleyiniz.*
    - g *ile oluşturulan iterasyonların hiçbir* $x_0 \neq 0$ *için* $r = 0$ *sabit noktasına yakınsamayacağını uygun* x_0 *başlangıç değerleri seçerek gözlemleyiniz.*

12. $f(x) = (x + 2)\exp(-x)$ *fonksiyonunun sıfıryerini Newton yöntemiyle belirleyebilmek için en büyük* $x_0 > -2$ *başlangıç tahmini ne olabilir?*

13. $f(x) = x - sinh(x)$ *fonksiyonu verilsin. Uygun bir* x_0 *başlangıç noktası seçerek, sırasıyla* $m = 1; 2$ *ve* 3 *için*

$$x_{n+1} = x_n - mf(x_n)/f'(x_n), n = 0, 1, 2, ...$$

Newton-Raphson iterasyonlarını $|x_{n+1} - x_n| < 10^{-5}$ *sonuçlandırma kriteri sağlanıncaya kadar hesaplayınız. Her bir* m *için gerekli iterasyon sayısını not ediniz? Hangi* m *değeri için en hızlı yakınsak diziyi elde ettiniz? Neden?*

14. $f(x) = x^2 - x - 1$ *fonksiyonu ve* $x_0 = \sqrt{2}$ *başlangıç noktası ve basit sıfıryerleri için Newton-Raphson yönteminin Örnek 6.1 de belirtildiği gibi* 5 *adımda yakınsarken,* $m = 2$ *ile katlı sıfyerleri için düzenlenen versiyonun* 2.0000 *ve* 1.3333 *değerleri arasında alternatif olarak salınım yaptığını gözlemleyiniz.*

15. *Eğer* r *noktası* f *fonksiyonunun basit sıfıryeri ve büküm noktası değilse, bu taktirde* f *ile oluşturulan* g *Newton-Raphson iterasyon fonksiyonunun* r *noktasında bir ekstremuma sahip olduğunu ispatlayınız.*

16. *Bu bölümde verilen Newtons isimli programı(Program 6.2) Örnek 6.8 de verilen denklem sistemi ve* $(x0, y0) = (2, 1)$ *ile çalıştırarak elde edilen sonuçların doğruluğunu kontrol ediniz. İterasyonların sistemin gerçek çözümüne yakınsadığını gözlemleyiniz.*

17. *Aşağıdaki nonlineer sistemlerin grafiklerini çizerek, kaç adet reel çözüme sahip olduklarını tahmin ediniz.*

- *(a)*

$$\begin{aligned} y^2 - 4x &= 1 \\ y^2 + 4x &= 1 \end{aligned}$$

- *(b)*

$$\begin{aligned} (x - 3/2)^2 + (y - 5/2)^2 &= 1/2 \\ (x - 3/2)^2 + (y - 1)^2 &= 5/4 \end{aligned}$$

- *(c)*

$$\begin{aligned} x^2/9 + y^2/4 &= 1 \\ x^2/4 + y^2/9 &= 1 \end{aligned}$$

- *(d)*

$$\begin{aligned} x^2/9 + y^2/4 &= 1 \\ x^2/4 - y^2/9 &= 1 \end{aligned}$$

18. *Soru 17(a) ya ait nonlineer sistem için* $(x_0, y_0) = (3, 3)$ *başlangıç noktası ile* (x_1, y_1) *ve* (x_2, y_2) *yaklaşımlarını Newton yöntemi yardımıyla hesaplayınız.*

19. *Soru 17(b) ait nonlineer sistem ve aşağıda verilen başlangıç değerlerin her birisi için* (x_1, y_1) *ve* (x_2, y_2) *yaklaşımlarını Newton yöntemi yardımıyla hesaplayınız.*

    - $(x_0, y_0) = (3, 3)$
    - $(x_0, y_0) = (-3, 3)$
    - $(x_0, y_0) = (-3, -3)$
    - $(x_0, y_0) = (3, -3)$

      Elde ettiğiniz yaklaşımlar, şekildeki grafiklerin arakesit noktalarına yakınsıyor mu?

20. *Soru 17(c) nin her bir reel çözümünü uygun başlangıç değerleri ve Program 6.2 ile belirleyiniz.*

6.5 Aralık üzerinde sıfıryerleri

6.5.1 Vektörel ikiye bölme yöntemi

Birinci bölümde incelenen klasik ikiye bölme yöntemi $[a, b]$ aralığında sürekli ve $f(a)f(b) < 0$ şartını sağlayan fonksiyonun söz konusu aralık içerisindeki tek bir sıfıryerini belirler. Ancak fonksiyon verilen aralığın uç noktalarında işaret değiştirmeyip, alt aralık veya aralıklarda işaret değiştirebilir. O halde verilen bir aralığın kullanıcı tarafından belirlenen uzunluktaki alt aralıkları taranarak işaret değişiminin gerçekleştiği alt aralıklar belirlenebilir. Bu bölümde amacımız

- fonksiyonun işaret değiştirdiği alt aralıkları bulmak,
- klasik ikiye bölme yöntemini eş zamanlı olarak her bir alt aralığa uygulayarak verilen keyfi bir $[a, b]$ aralığında yer alan tüm reel sıfıryerlerini belirlemektir.

Öncelikle birinci amaca yönelik aşağıdaki algoritmayı dikkate alalım:

Algoritma 6.1 İşaret değişim aralıklar algoritması

1. input: f,a,b,dx
2. A: aralık sol uç noktalar vektörünü boş küme olarak tanımla.
3. B: aralık sağ uç noktalar vektörünü boş küme olarak tanımla.
4. Tesadüfen bulunabilen sıfıryerleri saklamak için sıfırlar isimli boş bir küme tanımla.
5. sonuçlandırma kriteri olarak kullanılan test değişkenine 1 değerini ata.
6. Süreksizlik kontrolü için *Maxzipla* değişkenine 5 değerini ata.
7. test=1 olduğu sürece
   (a) f(a)=0 ise a yı sıfırlar dizisine aktar.
   (b) f(a+dx)=0 ise a+dx i sıfırlar dizisine aktar.
   (c) f(a)f(a+dx)<0 ve |f(a+dx)-f(a)|<*Maxzipla* ise a yı A dizisine ve $a + dx$ i de B dizisine aktar.
   (d) a= a+dx olarak al.
   (e) a>=b ise test=0 olarak al.
8. A ve B dizileri ile sıfırlar dizisini çağıran programa geri gönder.

Algoritma 6.1 için geliştirilen Program 6.3 aşağıda verilmektedir.

ÖRNEK 6.9. *Program 6.3 ile* $f(x) = cos(6\cos^{-1}(x))$ *Chebyshev*[4] *fonksiyonunun* $[-1, 1]$ *aralığında işaret değiştirdiği* 0.1 *uzunluklu alt aralıkları belirleyiniz.*

[4]Pafnuty Lvovich Chebyshev, 1821-1894, Rus matematikçi

```
% Sıfıryerlerini içeren aralık uc noktalar vektörü
% ve tesadüfen bulunan sıfıryerlerini
% çağıran programa  gönderir.
% A: sol uç noktalar
% B: sağ uç noktalar
%-----------------------------------------------
function [A,B,sifirlar]=ikibolaralik(f,a,b,dx)
test=1;
A=[];B=[];sifirlar=[];
while test
     if f(a)==0
         sifirlar=[sifirlar;a];
     end
     if f(a+dx)==0
         sifirlar=[sifirlar;a+dx];
     end
     if f(a)*f(a+dx)<0
         A=[A;a];B=[B;a+dx];
     end
     a=a+dx;
    if a>=b test=0;
    end
end
```

Program 6.3: İşaret değişim aralıklar uygulaması

$>>$ $f(x) = sin(5x)$ fonksiyonunun $[0, 4]$ aralığı içersinde sıfıryerlerini içeren $dx = 0.1$ uzunluklu alt aralıkları belirleyiniz.

```
>> [A,B,sifirlar]=ikibolaralik(f,0,4,0.1)
A =
0.6000 1.2000 1.8000 2.5000 3.1000 3.7000
B =
0.7000 1.3000 1.9000 2.6000 3.2000 3.8000
sifirlar = 0
```

O halde verilen fonksiyonun

$$[0.6, 0.7], [1.2, 1.3], [1.8, 1.9], [2.5, 2.6], [3.1, 3.2], [3.7, 3.8]$$

aralıklarının herbirinde enaz bir sıfıryeri vardır. Ayrıca $x = 0$ fonksiyonun bir sıfıryeridir.

İkiye bölme yöntemini eş zamanlı olarak yukarıda verilen alt aralıklara uygulayarak, herbir alt aralıktaki sıfıryerini belirlemek istiyoruz. Bu amaçla aşağıdaki algoritmayı uygulayalım

ÖRNEK 6.10. *$f(x) = \sin(5x)$ fonksiyonunun $[0, 4]$ aralığında yeralan sıfıryerlerini vektörel ikiye bölme yöntemi yardımıyla belirleyiniz.*

```
>> ikibolv(f,0,4)
```

0	0.628318530717962	1.256637061435919
1.884955592153875	2.513274122871833	3.141592653589795
...	...	...
8.168140899333460	8.796459430051419	9.424777960769381

Yukarıdaki tabloda verilen sonuçlarla gerçek değerler karşılaştırıldığında, elde edilen yaklaşımların virgülden sonra ondört basamağa kadar doğru olduğu anlaşılmaktadır.

ÖRNEK 6.11. *$f(x) = \tan(x)$ fonksiyonunun $[0, 10]$ aralığında yeralan sıfıryerlerini yöntemi yardımıyla belirleyiniz.*

```
>> ikibolv(f,0,10)
ans =0 3.141592653589784 6.283185307179581 9.424777960769376
```

elde ederiz.

Algoritma 6.2 Vektörel ikiye bölme yöntemi algoritması

1. input f, a, b, dx
2. Program 6.3 yi çağırmak suretiyle AB ile gösterilen aralık uç nokta vektörleri ile tesadüfen bulunabilen $sifirlar$ vektörünü oluştur.
3. $sifirlar$ vektörünü X ile gösterilen vektöre aktaralım. $A = AB(:,1); B = AB(:,2)$ olarak tanımla.
4. A ve X boş küme ise işaret değişim aralığıbelirlenemediği için programı sonlandır.
5. A boş küme ise X de varsa tesadüfi değerleri sıfıryeri olarak geri gönder.
6. $fark$ değişken değerini 1, eps sonuçlandırma kriteri değerini $1e-5$ olarak kabul et.
7. $fark$ değişkeninin sonsuz normu eps den büyük olduğu sürece aşağıdakileri tekrarla:
   (a) $C = (A+B)/2$ orta noktalar vektörünü hesapla.
   (b) $f(A).*f(C) < 0$ eşitsizliğini sağlayan A ve C vektörlerinin ii ile gösterilen indislerini belirle.
   (c) Eğer ii indis kümesi boştan farklı ise $B(ii) = C(ii)$ olarak değiş tir.
   (d) $f(A).*f(C) >= 0$ eşitsizliğini sağlayan A ve C vektörlerinin jj ile gösterilen indislerini belirle.
   (e) Eğer jj indis kümesi boştan farklı ise $A(jj) = C(jj)$ olarak değiştir.
   (f) $fark = |B - A|$ vektörünü tanımla.
   (g) $fark <= eps$ veya $|f(C)| < eps$ özelliğini sağlayan j_0 yakınsak aralık indislerini belirle.
   (h) $X = C(j_0)$ ile elde edilen sıfıryerlerini X vektörüne ata.
   (i) $fark > eps$ özelliğini sağlayan henüz yakınsamayan alt aralıkların j_1 indislerini belirle.
   (j) Eğer j_1 boştan farklı ise $A = A(j_1)$ ve $B = B(j_1)$ olarak değiştir.

İkiye bölme yönteminin bilinen skaler versiyonu verilen aralığın uç noktalarında fonksiyonun işaret değiştirmiş olmasını gerektirirken, yukarıda incelenen vektörel versiyon da fonksiyonun işaret değiştirdiği alt aralıkları belirleyerek, her bir alt aralıktaki sıfıryerini bulur. Ancak, fonksiyonun katlı sıfıryerlerini içeren aralıklarda işaret değiştirmeyebileceğini biliyoruz. Örneğin $f(x) = x^2$ fonksiyonu $[-1, 1]$ aralığında işaret değiştirmediği halde $x = 0$ noktasında iki katlı sıfıryerine sahiptir. Öte yandan $f(x) = x^3$ fonksiyonu da $x = 0$ da üç katlı sıfıryerine sahiptir ve aynı aralıkta işaret değiştirir.

Bu amaçla aralık uç noktalarında işaret değiştirmeyi gerektirmeyen iteratif yöntemlerin vektörel versiyonları çalışılabilir.

6.5.2 Vektörel *Newton* Yöntemi

Sabit nokta iterasyon yöntemleri bir disk içerisinde yer alan tüm sıfıryerlerini(reel ve kompleks) bulacak biçimde geliştirilebilirler. Özel olarak *Newton* yöntemi *MATLAB/Octave* vektör cebiri kullanılarak bu amaçla geliştirilebilir. Reel sıfıryerleri için uygun başlangıç vektör tahmini ile [7] yüksek lisans tez çalışmasında sunulmuştur.

Bu bölümde [7] de geliştirilen reel sıfıryerleri algoritmasını, sıfır merkezli diskteki tüm sıfıryerlerini belirleyecek biçimde geliştirerek Algoritma 6.3 ile aşağıda veriyoruz.

Algoritma 6.3 ya ait Program 6.4 bölüm alıştırmaları sonunda verilmektedir.

ÖRNEK 6.12. *$f(x) = x^3 - 1, x^5 - 1, x^7 - 1, x^9 - 1$ fonksiyonlarının tüm sıfıryerlerini yukarıdaki algoritma yardımıyla belirleyerek, sıfıryerlerini birim çember üzerinde gösteriniz.*

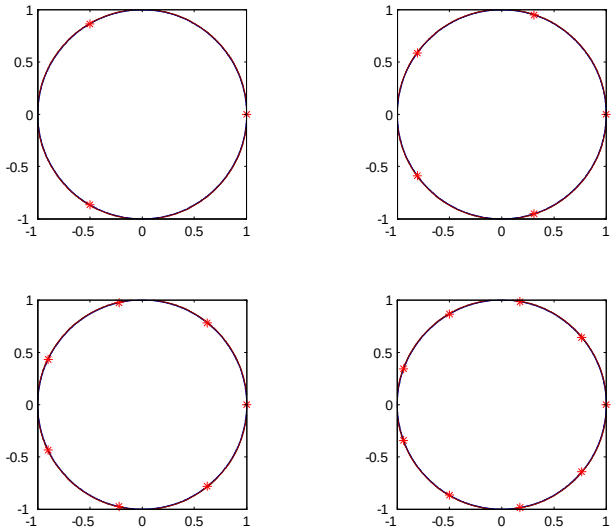
Çözüm.

Sırasıyla herbir fonksiyon için aşağıdaki program parçasını uygun grafik penceresinde çalıştıralım.

```
x=-1:0.01:1;
y=x;
[X,Y]=meshgrid(x,y);
Z=X.^2+Y.^2-1;
subplot(2,2,1);
```

```
contour(X,Y,Z,[0 0.01]);
hold on;
f=inline('x.^3-1');
df=inline('3*x.^2');
Xff=cvnewton(f,df,2,1);
plot(Xff,'*r');axis('square');
```

Elde ettiğimiz ekran çıktısı Şekil 6.5 te sunulmaktadır .



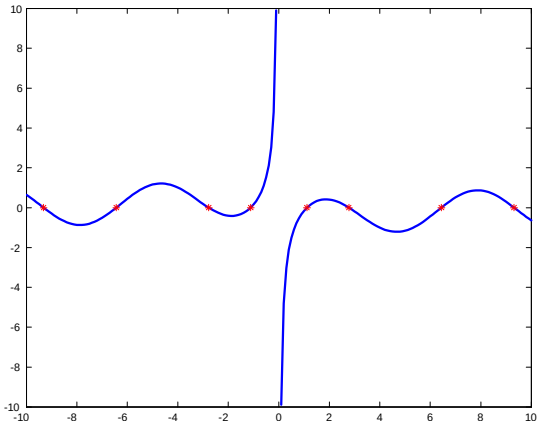
Şekil 6.5: $x^n - 1$ in $n = 3, 5, 7$ ve 9 için sıfıryerleri.

ÖRNEK 6.13. *$f(x) = \sin(x) - 1/x$ fonksiyonunun $[-10, 10]$ aralığındaki sıfıryerlerini Program 6.4 ile belirleyiniz.*

Çözüm. *>> f=inline('sin(x)-1./x')*
>> df=inline('cos(x)+1./x.^2')
>> Xff=cvNewton(f,df,10,1)
Xff = -9.3172 -6.4391 -2.7726 -1.1142 1.1142 2.7726 6.4391 9.3172

Elde edilen sıfıryerleri ve fonksiyon grafiği Şekil 6.6 de gösterilmektedir.

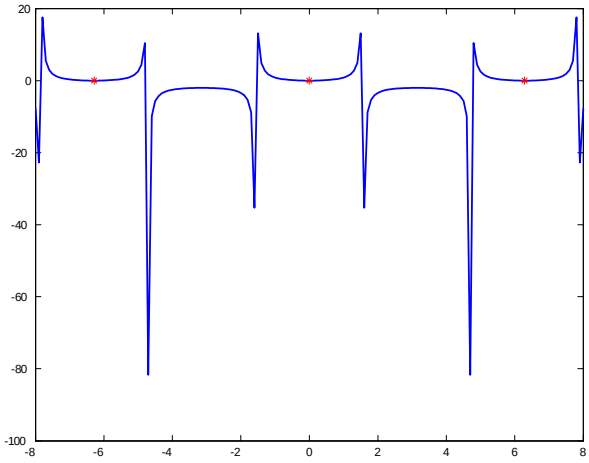
Sıfıryerini içeren aralıkta fonksiyonun işaret değiştirmemesi durumunda vektörel ikiye bölme yöntemi yerine vektörel *Newton* yönteminin kullanılması uygundur. Aşağıdaki örnekte fonksiyonun sıfıryerlerinde işaret değiştirmediğine dikkat ediniz.



Şekil 6.6: $f(x) = sin(x) - 1/x$ fonksiyonu grafiği ve sıfıryerleri(*)

ÖRNEK 6.14. $f(x) = \sec(x) - 1$ *fonksiyonunun* $[-10, 10]$ *aralığındaki sıfıryerlerini Program 6.4 ile belirleyiniz.*

Çözüm. >>f=inline('sec(x)-1')
>>df=inline('sec(x).*tan(x)')
>>Xff=cvNewton(f,df,10)
>>Xff =-6.2832 -0.0000 6.2832



Şekil 6.7: sec(x)-1 fonksiyonu ve sıfıryerleri

Alıştırmalar 6.2.

1. *Vektörel ikiye bölme yöntemine ait algoritma6.2 için uygun program hazırlayarak aşağıda verilen fonksiyonların karşılarında belirtilen aralıklardaki sıfıryerlerini belirleyiniz.*

   (a) $f(x) = x^3 + 3x^2 - 2x - 1, [-5, 5]$
   (b) $f(x) = \cos(x) - 1/x^2, [-8, 8]$
   (c) $f(x) = e^{-x^2} - \sin(x), [-7, 7]$
   (d) $f(x) = \tan(x) - \cos(x), [-10, 10]$

2. *Yukarıda verilen 6.3 algoritmasına ait cvNewton isimli Program 6.4 aşağıda verilmiştir. cvNewton programını yukarıda bölüm sonunda verilen çözümlü örnekler üzerinde test yapınız. Aynı sonuçları elde ediyor musunuz?*

3. *Soru 2 de incelediğiniz programı, Soru 1 de verilen örnekler üzerinde de test yapınız. Vektörel ikiyebölme yöntemi ile aynı sonuçları elde ediyor musunuz?*

4. *Proje(Polinomlar için Vektörel Newton Yöntemi): Fonksiyon ve türevini kullanıcıdan alarak belirtilen yarıçaplı disk içerisindeki tüm sıfıryerlerini belirleyen 6.3 algoritmasını, verilen polinomun tüm sıfıryerlerini bulacak biçimde ve CVNewtonp ismiyle geliştiriniz. Kullanıcının sadece polinomun katsayılar vektörünün girmesi yeterli olmalıdır.(İpucu: Öncelikle verilen bilgiden hareketle sıfıryerlerini içeren disk yarıçapını ilgili teoriyi araştırarak tahmin ediniz. Ayrıca türev polinomunun katsayılar vektörünü, verilen polinom katsayılar vektörü yardımıyla belirleyiniz.) Test sonuçlarınızı MATLAB/OCTAVE roots fonksiyonu ile elde edebileceğiniz sonuçlarla karşılaştırınız.*

Algoritma 6.3 Disk içersindeki tüm kökler için Vektörel Newton Algoritması

1. input: $f, df, r, secenek, dx, secenek = 0$:reel sıfıryerleri,$secenek = 1$:tüm sıfıryerleri
2. varsayılan değerler: $secenek = 0, dx = 0.1$;
3. Reel sıfıryerler için $X0 = -r : dx : r$;
4. Tüm sıfıryerleri için $X = [-r : dx : r]x[-r : dx : r]; Y = X; Z = X + iY$;
5. Z matrisinin satırlarını uç uca ekleyerek $X0$ başlangıç vektörü oluştur;
6. $Xf \leftarrow f(X0) = 0$;
7. $sayac = 0; Maxsayac = 15; carp = 1e10; eps = 1E - 14$;
8. $Maxsayac$ değerine ulaşılana kadar aşağıdadakileri tekrarla;
   (a) $df(X0)$ vektörünü hesapla ve $df(X0) \neq 0$ olan $i0$ indislerini belirle;
   (b) $X0 \leftarrow X0(i0)$;
   (c) $X1 \leftarrow X0 - f(X0)./df(X0)$;
   (d) $|X1| < r$ eşitsizliğini sağlayan ii indislerini belirle;
   (e) $X0 \leftarrow X0(ii), X1 \leftarrow X1(ii)$;
   (f) $fark \leftarrow |X1 - X0|$;
   (g) $fark \leq eps$ eşitsizliğini sağlayan $j0$ indislerini belirle;
   (h) $fark > eps$ eşitsizliğini sağlayan $j1$ indislerini belirle;
   (i) $sayac \leftarrow sayac + 1$;
   (j) $Xf \leftarrow X0(j0)$,yakınsayan bileşenleri Xf vektörüne yığ;
   (k) $X0 \leftarrow X1(j1)$,yakınsamayan bileşenleri güncel başlangıç vektörü olarak al;
   (l) $X0$ boş ise mevcut Xff i kullanıcıya ilet;
   (m) $secenek = 1$ ise
      i. reel ve sanal kısımları virgülden sonra on basamağa kadar yuvarla;
      ii. faklı bileşenleri belirle ve Xff vektöründe biriktir.
   (n) $secenek = 0$ ise
      i. yakınsak bileşenleri Xff vektöründe biriktir;
      ii. Xff vektörünü sırala;
      iii. ardışık elemanlar arasındaki farkı da dan büyük olanları farklı kabul et.

Karadeniz Teknik Üniversitesi Matematik Bölümü

```
function Xff=cvnewton(f,df,r,secenek,dx)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Vektörel Newton B(0,r) deki komplex/reel kökleri bulur
% secenek girilmezse reel sıfıryerlerini bulur.
% secenek=1 ise tüm sıfıryerlerini bulur.
% ec, Mayıs, 2015
% Bu program [7] de verilen programın geliştirilmiş versiyonudur.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

carp=1e6;   %kompleks kökler için yuvarlama parametresi
sayac=0;     %sayac değişkeni
eps=1E-14;   %yakınsama kontrol parameteresi
maxsayac=25; % maximum iterasyon sayısı

if nargin==3 secenek=0;dx=0.1;
end;
if nargin==4   dx=0.1;
end
  if secenek==0
     X0=-r:dx:r;X0=X0'; %reel sıfıryerleri için başlangıç
   elseif secenek==1    % tahmin vektörü
    x=-r:dx:r;y=j*x;n=length(x);n2=n*n;
    [X,Y]=meshgrid(x,y);
    Z=X+Y;
    X0=reshape(Z,n2,1); % tüm sıfıryerleri için başlangıç
                        % tahmin vektörü(reel ve kompleks)
  else error('kullanım: Xf=cvnewton(f,df,r,secenek,dx)');
  end
test=sayac<maxsayac;    % döngü değişkeninin ilk değeri 1 e eşit
ii=find(f(X0)==0);      % tesadüfen bulunabilen sıfıryerleri
Xf=X0(ii);              % sıfıryerlerini saklayan vektörel değişken
while test
    df0=df(X0);        %türevin X0 daki değerler vektörü
    i0=find(df0~=0);   % türevin sıfırdan farklı olduğu indisler
    X0=X0(i0);         %turevi sıfırdan farklı başlangıç değerleri
    X1=X0-f(X0)./df(X0);  %Vektörel Newton adımı

    ii=find(abs(X1)<r);  % r yarıçaplı bölgedeki sıfıryerleri indisleri
    X0=X0(ii);
    X1=X1(ii);
    fark=abs(X1-X0);     %yaklaşımlar arası fark
    j0=find(fark<=eps);  %yakınsayan bileşenlerin indisleri
    j1=find(fark>eps);   %yakınsamayan bileşenlerin indisleri
    sayac=sayac+1;       %döngü değişken değer artırımı
   if length(j0)>0
       Xf=[Xf;X1(j0)];   % yakınsayan bileşenleri X vektörüne diz
   end
```

Karadeniz Teknik Matematik, ercan@ktu.edu.tr

```
   if length(j1)>0
       X0=X1(j1);       % yakınsamayan bileşenlerle devam et
   end

    if isempty(X0)
        break;  %bölgede yakınsamayan bileşen yoksa döngüden çık
    end
    test=sayac<maxsayac;  % maxsayac ulaşılana kadar işlemleri tekrarla
end
if secenek==1
  Xfr=round(carp*real(Xf))/carp; %reel kısımları yuvarla
  Xfs=round(carp*imag(Xf))/carp; % sanal kısımları yuvarla
  Xfy=round(carp*Xfr)/carp+i*round(carp*Xfs)/carp; %yuvarlanmış değerler
  Xff=unique(Xfy);  %farklı olan yuvarlanmış değerleri belirle
 else
  ii=find(abs(f(Xf))<1e-5); % yakınsak bileşen indisleri
  if length(ii)>0
  Xf=Xf(ii);
  Xf=sort(Xf); %sırala
  Xff(1)=Xf(1);n=length(Xf)-1;
 for ii=2:n
    if abs(Xf(ii+1)-Xf(ii))>dx
        Xff=[Xff;Xf(ii+1)]; %farklı olanları seç
    end
 end
  else display('No reel zeros');
  end
end
```

Program 6.4: Geliştirilmiş Vektörel Newton yöntemi uygulaması

Kaynaklar

[1] Atkinson, K. An Introduction to Numerical Analysis, John Wiley & Sons, 1988.

[2] Coşkun, E. OCTAVE ile Sayısal Hesaplama ve Kodlama(URL:erhancoskun.com.tr).

[3] Coşkun, E. Maxima ile Sembolik Hesaplama ve Kodlama(URL:erhancoskun.com.tr).

[4] Edwards & Penney, çeviri ed. Akın, Ö., Diferensiyel Denklemler ve Sınır Değer Problemleri, Palme Yayıncılık, 2006.

[5] MATLAB, Mathworks(*URL:mathworks.com).*

[6] Maxima, GNU özgür yazılım(*URL:maxima.sourceforge.net*).

[7] Memoğlu, M., Vektörel Sıfıryeri ve ektremum nokta belirleme algoritmaları, Yüksek Lisans Tez Çalışması, KTÜ, 2012.

[8] OCTAVE, GNU özgür yazılım(*URL:OCTAVE.sourceforge.net*).

[9] Press, H. W. ve ark., Numerical Recipes in C, Cambridge University Press, 1988.

[10] Stoer, J., Bulirsh, R., Introduction to Numerical Analysis, Springer-Verlag, 1976.

[11] Strang, G., Introduction to Applied Mathematics, Wellesley-Cambridge, 1986.

[12] S. W., Warren, Zill, D. G., Calculus: Early Transcendentals, Çeviri: Matematik Cilt I, II(Çeviri editörü İsmail Naci Cangül), Nobel Akademik Yayıncılık, 2010.