SAYISAL ANALİZ

Dr. Öğr. ÜYESİ Abdullah SEVİN

SAYISAL ANALİZ

DENKLEM ÇÖZÜMLERİ

İÇİNDEKİLER

Doğrusal Olmayan Denklem Çözümleri

- Grafik Yöntemleri
- Kapalı Yöntemler
  - İkiye Bölme (Bisection) Yöntemi
  - Adım Küçülterek Köke Yaklaşma Yöntemi
  - Yer Değiştirme Yöntemi
- Açık Yöntemler
  - Basit Sabit Noktalı İterasyon
  - Newton-Raphson Yöntemi
  - Kiriş (Secant) Yöntemi

Denklem Çözümleri

- **Problemlerin çözümünde ve sistemlere ait bağımlı değişkenlerin tahmin edilmesinde kullanılırlar.**
- **Denklemler mühendislikte tasarımda kullanılır.**
- **Sayısal analizdeki matematiksel modelleme aşaması denklemler ve denklem çözümlerinden oluşur.**
- **Örnek: Bir paraşütçünün düşme hızının hesabı (Newton 2. yasası)**

$$v = \frac{g\,m}{c}\left(1 - e^{-(c/m)t}\right)$$

Denklem Çözümleri

- **Denklemler ikiye ayrılır**

❶ Doğrusal (Lineer) Denklemler

- **Bilinmeyen bir başlangıç değeri içerir,açık veya gizli olabilir.**
- **Bilinmeyen parametrelerdeki doğrusal değişiklikler yine bilinmeyen parametreli fonksiyona işaret eder.**

❷ Doğrusal Olmayan (Non-Lineer) Denklemler

- **Üssü birden farklı bir değere sahip olan ve/veya doğrusal olmayan fonksiyonlar içeren denklemlerdir.**

Denklem Çözümleri

- **Örnek: Bir paraşütçünün düşme hızının hesabı (Newton 2. yasası)**

$$v = \frac{g\,m}{c}\left(1 - e^{-(c/m)t}\right)$$

v: hız (diğer zorlayıcı kuvvetlere, parametrelere ve bağımsız değişkenlere bağlı olarak değişen bir bağımlı değişkendir.
t: zaman (bağımsız değişken)
g: yer çekimi sabiti (zorlayıcı kuvvet)
c: havanın direnç katsayısı (sistemin fiziksel özelliği)
m: kütle (sistemin fiziksel özelliği)

- **Diğer parametreler bilinirse, paraşütçünün hızını, zamana bağlı olarak hesaplamak (v=f(t)) kolaydır.**
- **c bilinmediğinde analitik çözümü yok!**
- **Çözüm sayısal analiz yöntemi kullanmak (f(c)=0)**

$$f(c) = \frac{g\,m}{c}\left(1 - e^{-(c/m)t}\right) - v$$

- **Fonksiyonu sıfır yapan değer (kök), c'ye tekrar tekrar değerler verilerek, grafik veya diğer sayısal yöntemlerle bulunur.**
- **Denklemlerin sayısal olarak çözümleri de diğer problem çözümleri gibi çoğunlukla yinelemeli (iteratif) yöntemlerle yapılır.**

Sayısal Analizde Denklem Köklerini Bulmada İzlenecek Yol

❶ **Denklem köklerini aramaya belirli bir başlangıç değeri ya da değer aralığından başlanır.**

- **Kökü aramaya doğru bir noktadan başlamak çözüme ulaşmayı hızlandıracaktır.**

❷ **Fonksiyonun girişine değerler vererek, fonksiyonun çıkışı gözlemlenir.**

- **Fonksiyonun çıkışını gözlemlemenin kolay yolu, fonksiyonun grafiğini çizdirmektir.**
- **Grafik, köke yakın aralığı hızlı ve kolay tespit etmeye sağlar.**
- **Kökü aramaya uygun yerden başlamayı sağlar.**

Grafik Yöntemleri

- Sayısal analiz ile denklem köklerini **hızlı** ve **kolay** bulmayı sağlayan bir yöntemdir.

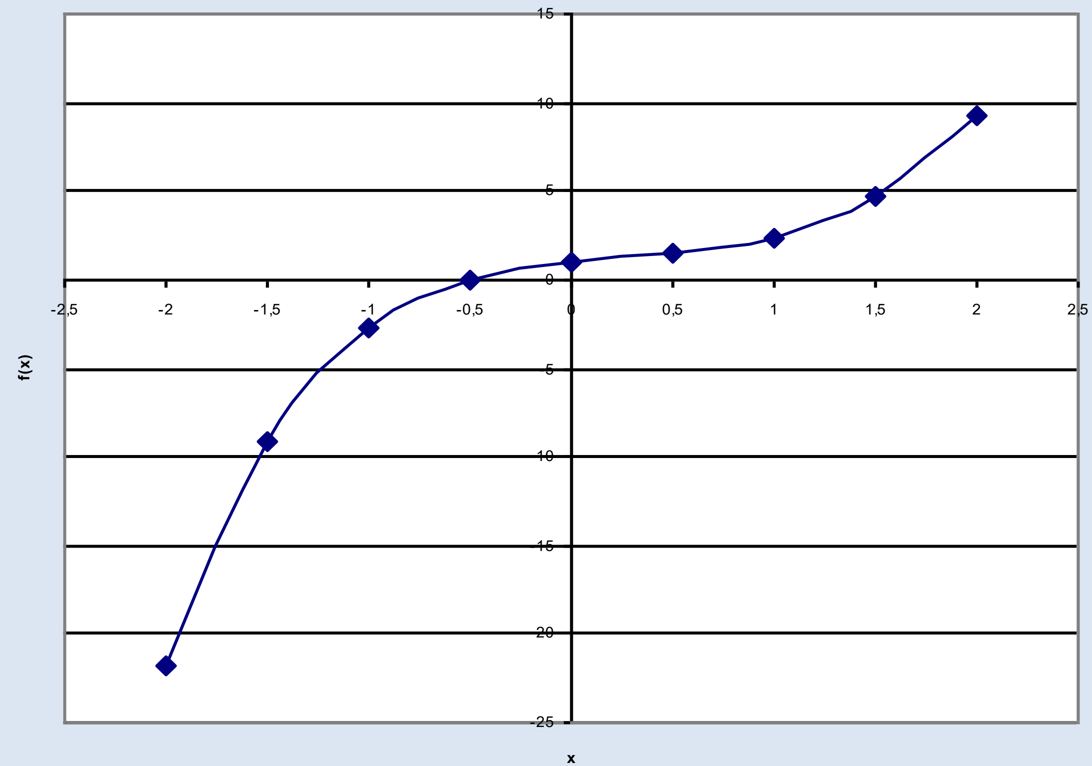
- Karmaşık denklem/problemlerin yaklaşık (kabaca) çözümlenmesini sağlar.

- Grafiksel yöntemlerin dezavantajları 
  1. Hassas çözüm elde edilemez
  2. Bilgisayar kullanmadan grafik çizmek uzun zaman alır
  3. Çoğunlukla 3 ya da daha düşük bilinmeyenli denklem çözümü için uygundur.

Grafik Yöntemleri

Örnek: $f(x) = xe^{-x} + x^3 + 1$ fonksiyonunun kökünü, grafik yöntemi ile yaklaşık olarak bulunuz?

x	f(x)
-2	-21,7781122
-1,5	-9,097533606
-1	-2,718281828
-0,5	0,050639365
0	1
0,5	1,42826533
1	2,367879441
1,5	4,70969524
2	9,270670566

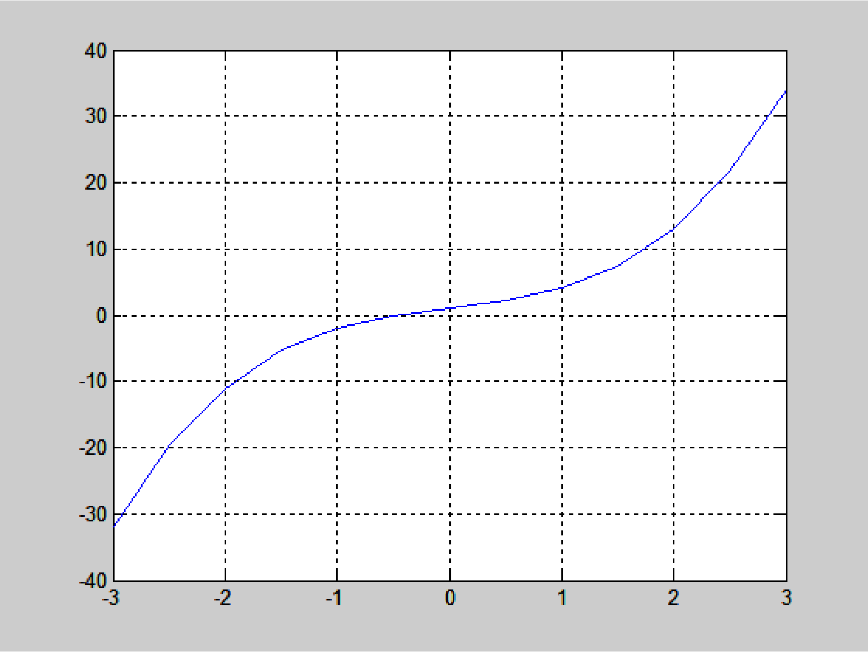


Grafik Yöntemleri

Örnek: $f(x) = x^3 + 2x + 1$ **fonksiyonunun kökünü, matlab programında çizdireceğiniz grafik üzerinden kabaca bulunuz?**

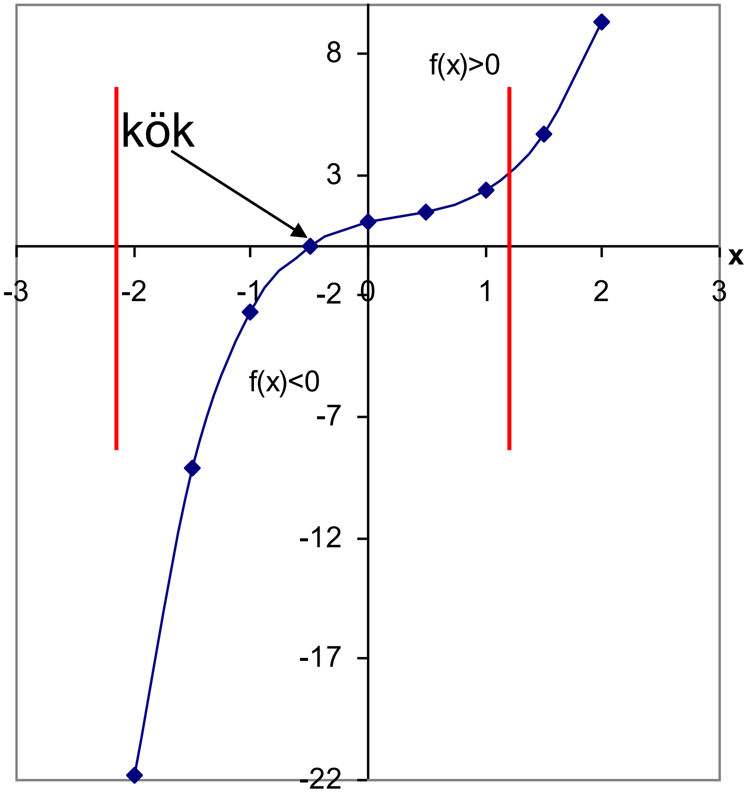
program.m

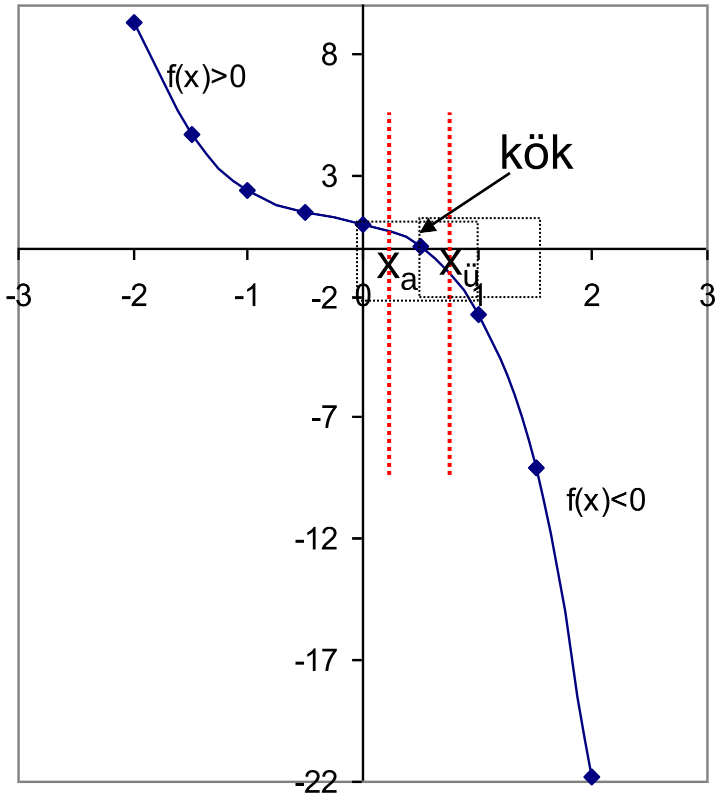
```
%  f(x) = x^3 + 2x + 1 denkleminin grafik yöntemi ile çözümü
x = -3:0.5:3;     % -3 < x < 3 aralığı
f = x.^3+2.*x+1;
plot(x,f)          % grafiği çizdir
grid on
```



Denklem Çözümünde Kapalı Yöntemler

- **Fonksiyonlar kök civarında işaret değiştirdikleri** için, ***kökü sağından ve solundan*** **kıskaca alarak bu aralığı gittikçe daraltıp köke ulaşmak mümkündür. Bunun için iki tane başlangıç değeri belirlemek gerekir.**
- **Kökün, bu iki değerin arasındaki kapalı bölgede olduğu bu yöntemlere** ***kapalı yöntemler*** **adı verilir.**

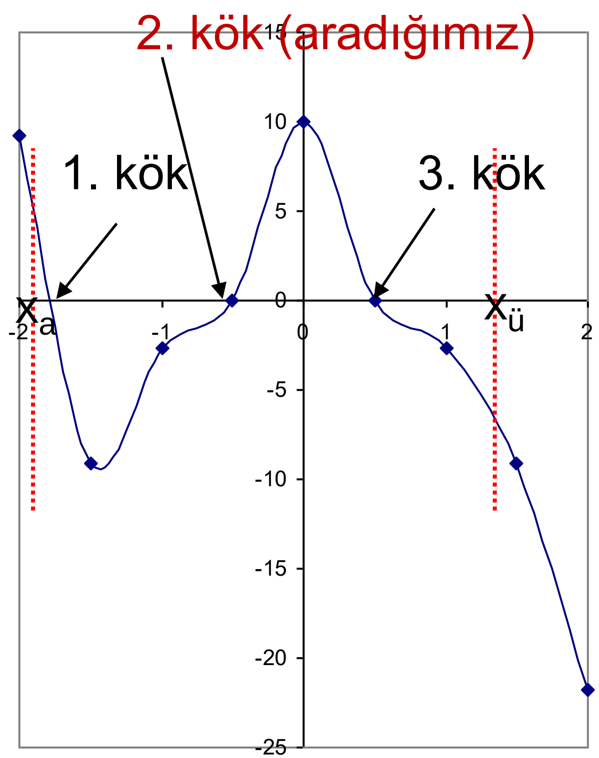




Serhat Yılmaz'ın Sunusundan Alınmıştır.

Denklem Çözümünde Kapalı Yöntemler

- **Doğru kökü hızlı ve sağlıklı olarak bulmak için, arada başka bir kök olmaması için aralık mümkün olduğunca dar seçilmelidir.**



İkiye Bölme (Yarılama, Bisection) Yöntemi

- Denklem çözümünde kapalı yöntemlerin bir türü olan *Bisection*, ikiye bölme ya da yarılama olarak ta adlandırılmaktadır.
- *Bisection*, sürekli bir fonksiyonun bir sıfırının (kökünün) bulunması için kullanılan sistematik bir tarama tekniğidir.
- Tekrarlama (tarama) yöntemlerinin en basit ve en anlaşılırıdır.
- Kökün bulunduğu aralığı yarılayarak (ikiye bölerek) daraltma prensibine dayanır.
  - Bu yöntem, içerisinde bir sıfır bulunan bir aralığın öncelikle tespitine dayanır.
  - Aralık sonunda fonksiyon zıt işarete sahiptir.
  - Sonra aralık iki eşit alt aralığa bölünür ve hangi aralığın bir sıfır değeri içerdiğine bakılır.
  - Sıfır içeren alt aralıklarda hesaplamalara devam edilir.
- Dezavantajı, yavaş yakınsaması ve bazen tam olarak çalışmaması.

İkiye Bölme (Yarılama, Bisection) Yöntemi

- Bir f(x) fonksiyonu, [x_a,$x_ü$] aralığında bir sıfır noktasına (köke) sahip olduğunu varsayalım.

❶ İlk olarak, f(x) fonksiyonunun belirtilen aralıkta kökü olup olmadığı [$f(x_a)* f(x_ü) < 0$] kontrol edilir. Şart sağlıyorsa kök vardır. Çünkü fonksiyonlar zıt işaretlidir.

- [$f(x_a)* f(x_ü) > 0$] ise kök yoktur.
- [$f(x_a)* f(x_ü) = 0$] ise kök x_a ya da $x_ü$

❷ İlk iterasyonda, belirtilen fonksiyon aralığının orta noktası tespit edilir.

$$x_o = \frac{x_a + x_ü}{2}$$

❸ Sıfır noktası [x_a, x_o] ya da [x_o, $x_ü$] aralığından birisinde olmalıdır

- $f(x_a)* f(x_o) < 0$ ise kök [x_a, x_o] aralığında
- $f(x_o)* f(x_ü) < 0$ ise kök [x_o, $x_ü$] aralığında

❹ Bir sonraki iterasyonda kök yeni aralıkta aranır ve 2. adımdan itibaren işlemler tekrarlanır.

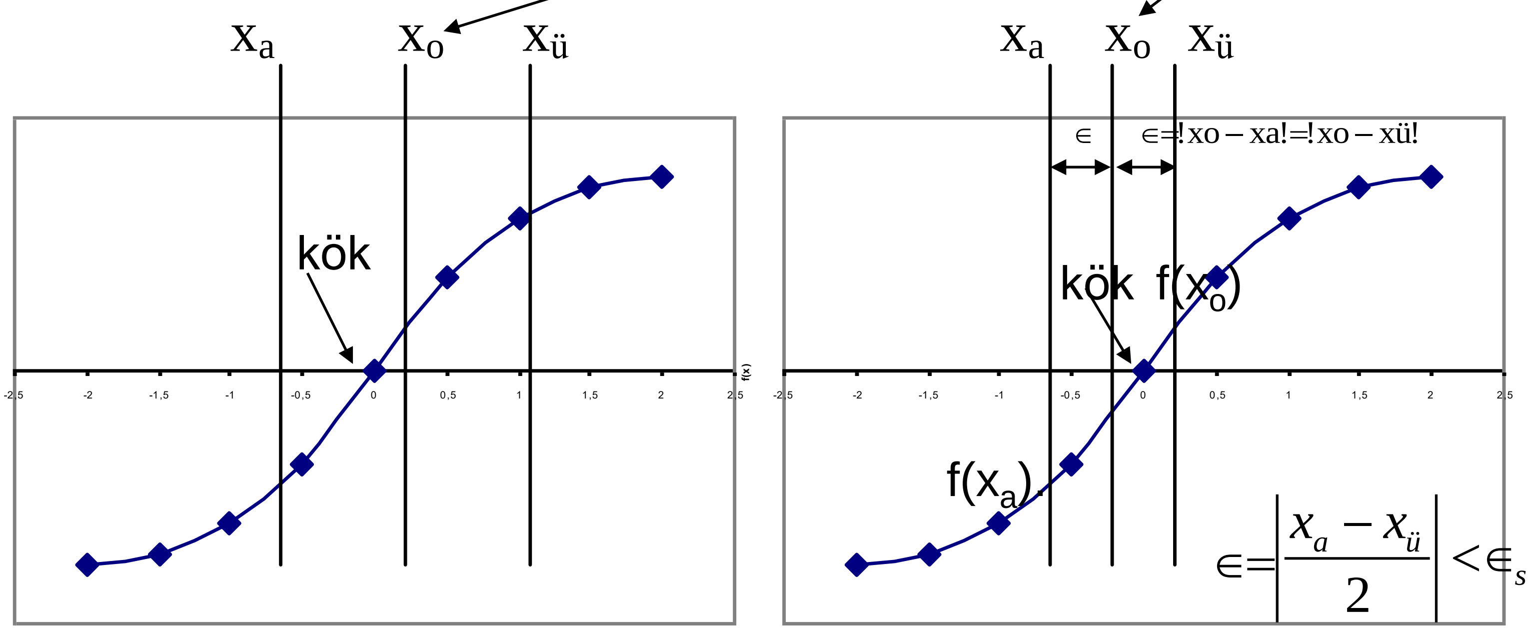
- Tekrarlama işlemi $\frac{|x_a - x_ü|}{2} < \varepsilon_s$ şartı sağlanana kadar devam eder.

İkiye Bölme (Yarılama, Bisection) Yöntemi

$$x_o = \frac{x_a + x_ü}{2}$$

Güncellenecek sınır

- $f(x_a).f(x_o) < 0$ x_a ile x_o farklı bölgelerde xü(yeni)=xo
- $f(x_a).f(x_o) > 0$ x_a ile x_o aynı bölgelerde xa(yeni)=xo



Kök, x_a, x_o arasında

Kök, x_o, $x_ü$ arasında

İkiye Bölme (Yarılama, Bisection) Yöntemi

❖ **Örnek :** **$f(x) = x.e^{-x}+x^3+1$** fonksiyonunun kökünü $\varepsilon_s = 1*10^{-6}$ duyarlılıkla bulalım,

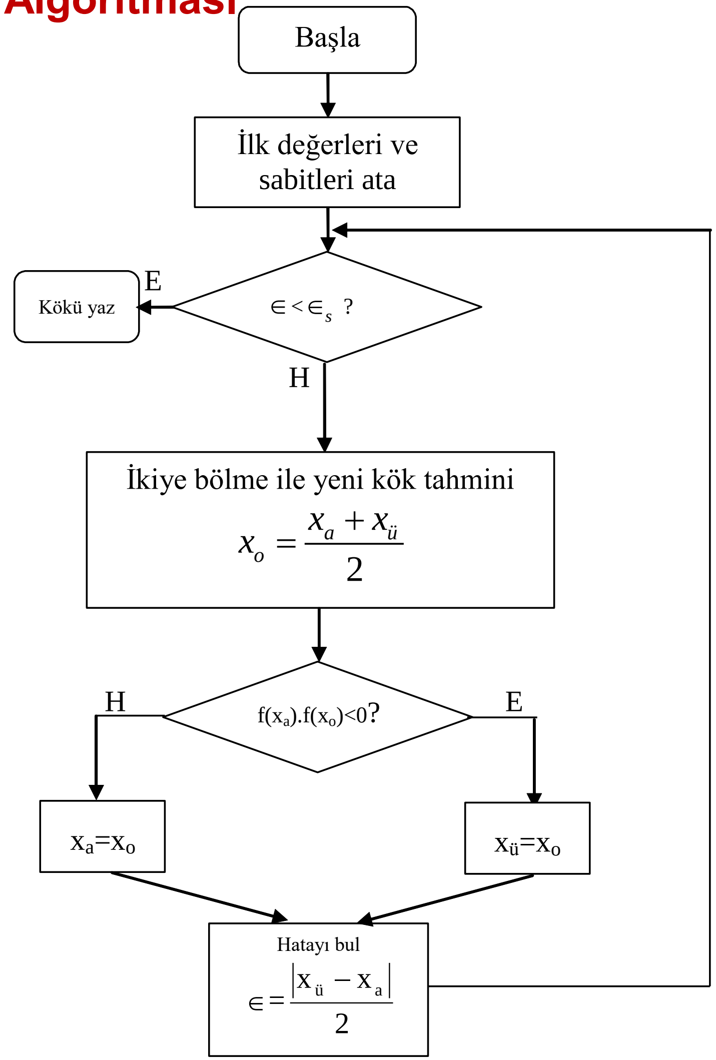
Not: Grafik yönteminde, [-1,0] aralığı için kabaca sonuç x=-0.515438

n	x_a	$x_ü$	x_o	$f(x_a).f(x_o)$	$\in = \left\lvert \frac{x_a - x_ü}{2} \right\rvert$
1	-1.000000	0.000000	-0.500000	-	0.500000
2	-1.000000	-0.500000	-0.750000	+	0.250000
3	-0.750000	-0.500000	-0.625000	+	0.125000
4	-0.625000	-0.500000	-0.562500	+	0.062500
5	-0.562500	-0.500000	-0.531250	+	0.031250
6	-0.531250	-0.500000	-0.515625	+	0.015625
7	-0.515625	-0.500000	-0.507813	-	0.007813
.	.	.	.	.	.
.	.	.	.	.	.
19	-0.515449	-0.515442	-0.515446	+	0.000004
20	-0.515446	-0.515442	-0.515444	-	0.000002

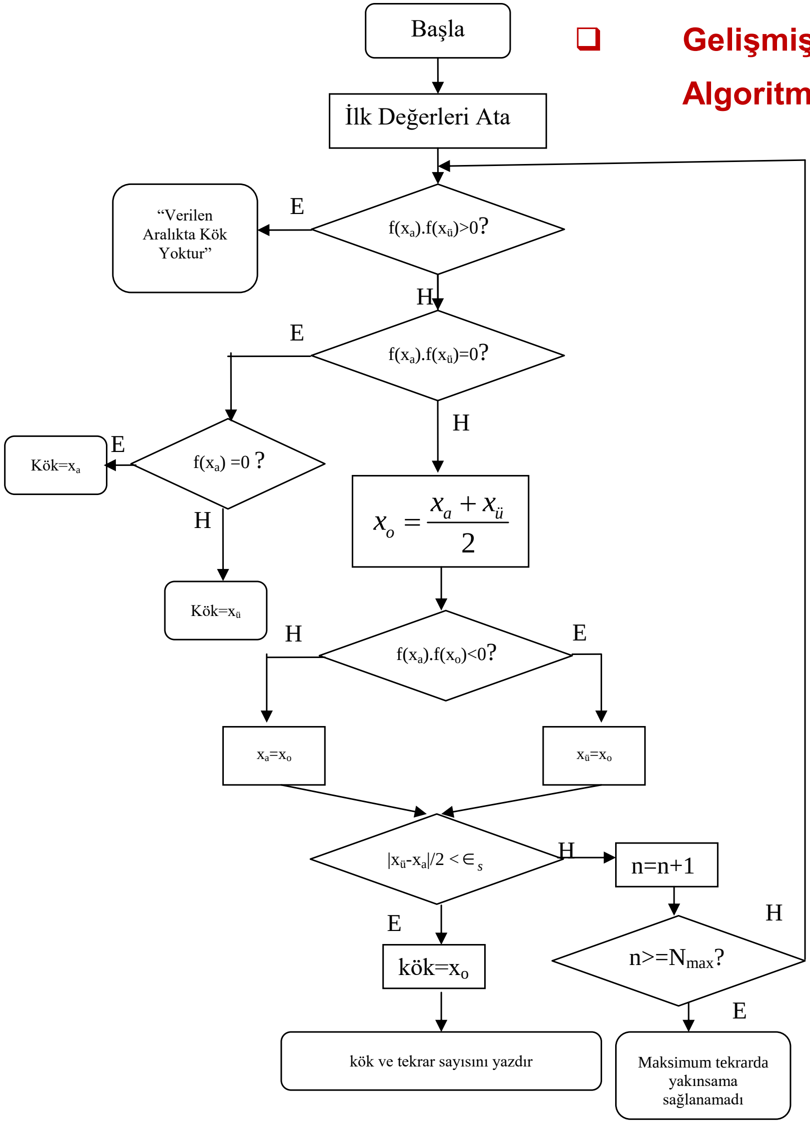
Serhat Yılmaz'ın Sunusundan Alınmıştır.

İkiye Bölme (Yarılama, Bisection) Yöntemi

- Algoritması



- Gelişmiş Algoritması



Serhat Yılmaz'ın Sunusundan Alınmıştır.

Bisection YÖNTEMİ MATLAB UYGULAMASI

```
xa=-1;xu=0;epsilon=1e-6;Nmax=100;n=0;
fxa=xa*exp(-xa)+xa^3+1;
fxu=xu*exp(-xu)+xu^3+1;

while n<Nmax
   if fxa*fxu > 0
      disp('Verilen aralıkta kök yoktur!!!')
      n=Nmax;
   elseif fxa*fxu == 0
      if fxa==0
         kok=xa;
      else
         kok=xu;
      end
   else
      xo=(xa+xu)/2;
      fxa=xa*exp(-xa)+xa^3+1;
      fxo=xo*exp(-xo)+xo^3+1;
```

```
      if fxa*fxo<0
         xu=xo;
      else
         xa=xo;
      end
   end
   if abs(xu-xa)/2<epsilon
      kok=xo;
      disp('Kok=')
      disp(kok)
      disp('Tekrar sayısı=')
      disp(n)
      n=Nmax;
   else
      n=n+1;
   end
end
```

İkiye Bölme (Yarılama, Bisection) Yöntemi

❖ **Örnek :** **f(x) = x² – 17** denkleminin **[4 , 5]** aralığında kökü olup olmadığını kontrol ederek, eğer var ise kök değerini **ikiye bölme** (bisection) **metodunu** kullanarak köke 2 adım yaklaşınız? Her adımda oluşan hatayı da hesaplayınız.

Not: Tüm değerler virgülden sonra 4 basamak alınacak.

❑ **Belirtilen aralıkta kök olup olmadığını kontrol edelim.**

$$f(x_a).f(x_{\ddot{u}})<0 \quad \Rightarrow \quad f(4).f(5)<0 \quad \Rightarrow \quad (-1).(8)<0 \quad \textit{kök}\ \text{var}$$

❶ **İterasyon**

$$x_0=\frac{x_a+x_{\ddot{u}}}{2}=\frac{4+5}{2}=4.5 \qquad \varepsilon=\frac{|x_a-x_{\ddot{u}}|}{2}=\frac{|4-5|}{2}=0.5$$

Kök hangi yarıda ?

$$f(x_a).f(x_o)<0 \quad \Rightarrow \quad f(4).f(4.5)<0 \quad \Rightarrow \quad x_{\ddot{u}}=4.5$$

❷ **İterasyon**

$$x_0=\frac{x_a+x_{\ddot{u}}}{2}=\frac{4+4.5}{2}=4.25 \qquad \varepsilon=\frac{|x_a-x_{\ddot{u}}|}{2}=\frac{|4-4.5|}{2}=0.25$$

Kök hangi yarıda ?

$$f(x_a).f(x_o)<0 \quad \Rightarrow \quad f(4).f(4.25)<0 \quad \Rightarrow \quad x_{\ddot{u}}=4.25$$

İkiye Bölme (Yarılama, Bisection) Yöntemi

❖ **Örnek :** **$f(x) = e^x - 4x$** denkleminin [**0 , 1**] aralığında kökü olup olmadığını kontrol ederek, eğer var ise kök değerini **yarılama** (bisection) **metodunu** kullanarak köke 3 adım yaklaşınız?

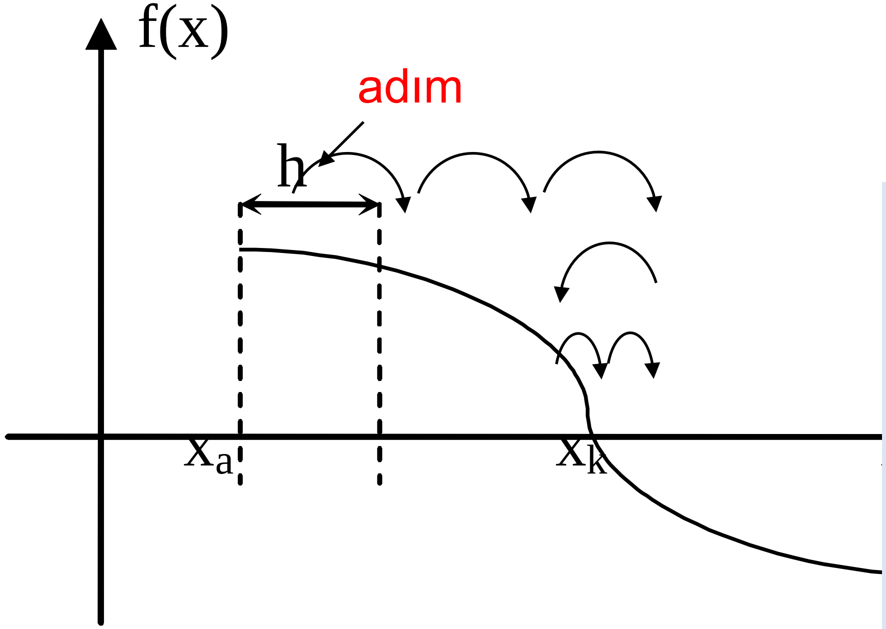
Not: Tüm değerler virgülden sonra 4 basamak alınacak.

Adım Küçülterek Köke Yaklaşma Yöntemi

- Ardışıl yaklaşım yöntemi olarak ta bilinir.
- Bir başlangıç değerinden başlanarak, adım adım (h, sabit mesafeler) köke yaklaşılır.
  - Önce büyük adımlar ile başlanır.
- [$f(x_a)* f(x_ü) > 0$] $\Rightarrow$ [$f(x)* f(x+h) > 0$] şartı sağlandığı sürece
  - Bir adım daha ilerlenir.
  - Adım büyüklüğünde (h) değişiklik yapılmaz.
- [$f(x)* f(x+h) < 0$] ise kök geçilmiştir.
  - En son kalınan başlangıç değerinden, adım küçülterek tekrar ilerlemeye devam edilir.
- Hata sınırlaması sağlanana kadar köke yaklaşmaya devam edilir.
  - $h < \varepsilon_s$

Adım Küçülterek Köke Yaklaşma Yöntemi

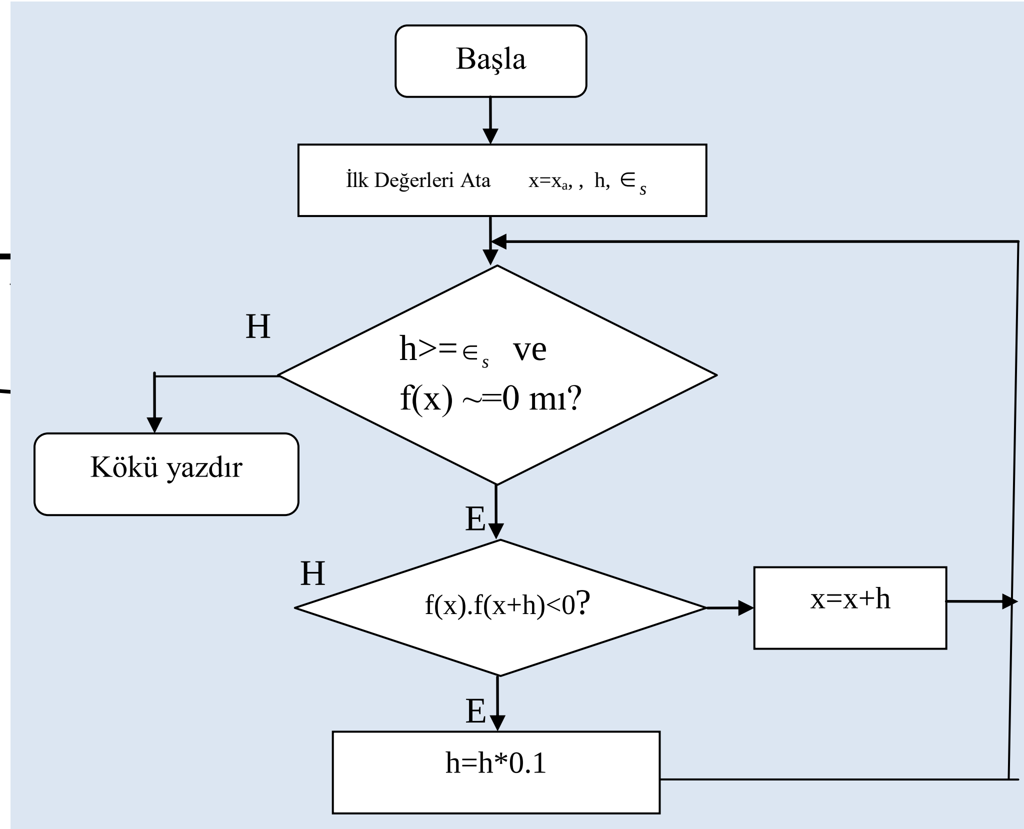
❑ Çalışması



$f(x).f(x+h) > 0 \longrightarrow$ $\boxed{x(yeni)=x+h}$

$f(x).f(x+h) < 0 \longrightarrow$ h(yeni)=h/10

❑ Algoritması



Adım Küçülterek Köke Yaklaşma Yöntemi

❖ **Örnek :** $f(x) = x^2 - 17$ denkleminin kökünü 4 değerinden ve 0.1 adım büyüklüğü ile başlayarak köke 5 adım (ya da hata sınırı 0.01 oluncaya kadar) yaklaşınız? Kökü geçtiğiniz her noktadan sonra adım büyüklüğünü **1/10** oranında küçültünüz.

Not: Tüm değerler virgülden sonra 2 basamak alınacak.

❶ **İterasyon : Belirtilen aralıkta kök olup olmadığını kontrol edelim.** **h=0.1 yeni $x_ü$ = 4 + 0.1=4.1**

$$f(x_a).f(x_ü) > 0 \quad \Rightarrow \quad f(4).f(4.1) > 0 \quad \Rightarrow \quad (-1).(-0.19) > 0 \quad \textit{kök yok}$$

❷ **İterasyon : Yeni aralıkta kök olup olmadığını kontrol edelim.** **h=0.1 yeni $x_ü$ = 4.1 + 0.1=4.2**

$$f(x_a).f(x_ü) < 0 \quad \Rightarrow \quad f(4.1).f(4.2) < 0 \quad \Rightarrow \quad (-0.81).(0.64) < 0 \quad \textit{kök} \text{ var}$$

4.1 ile 4.2 aralığında (işaret değişikliği sebebiyle) kök olduğu tespit edildiğinden kökü aramaya bir önceki kök değerinden (4.1) tekrar başlanacak ancak adım büyüklüğü h=0.1, 1/10 oranında küçültülecektir (h=0.01)

❸ **İterasyon : 4.1 değerinden 0.01 adım büyüklükleri ile kökü ara**

$$f(x_a).f(x_ü) > 0 \quad \Rightarrow \quad f(4.1).f(4.11) > 0 \quad \Rightarrow \quad (-0.81).(-0.11) > 0 \quad \textit{kök yok}$$

❹ **İterasyon : Yeni aralıkta kök olup olmadığını kontrol edelim.** **h=0.01 yeni $x_ü$ = 4.11 + 0.01=4.12**

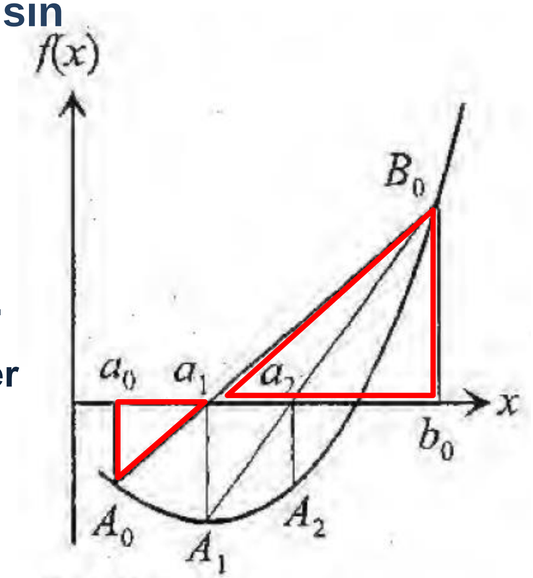
$$f(x_a).f(x_ü) > 0 \quad \Rightarrow \quad f(4.11).f(4.12) > 0 \quad \Rightarrow \quad (-0.11).(-0.03) > 0 \quad \textit{kök yok}$$

❺ **İterasyon : Yeni aralıkta kök olup olmadığını kontrol edelim.** **h=0.01 yeni $x_ü$ = 4.12 + 0.01=4.13**

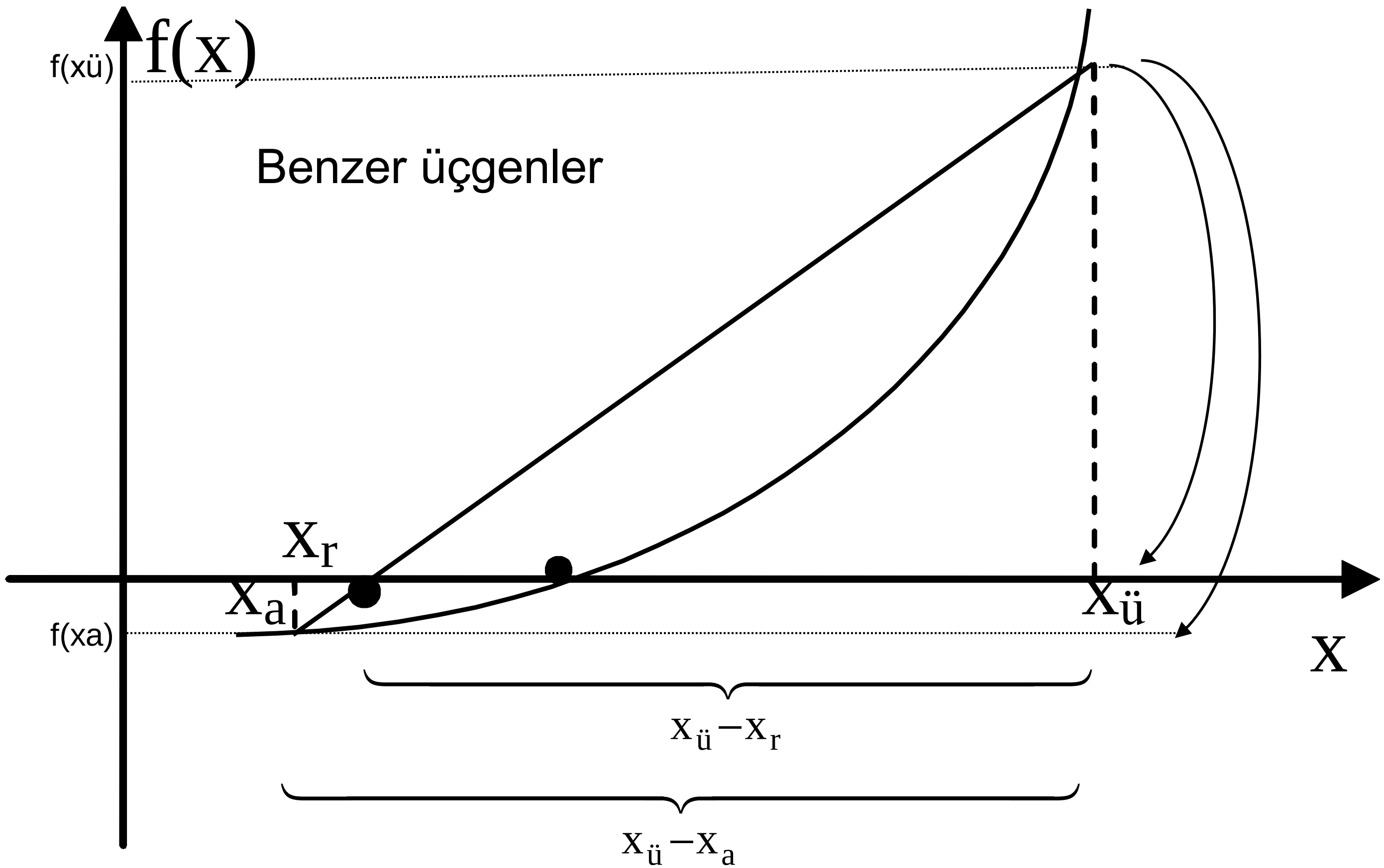
$$f(x_a).f(x_ü) < 0 \quad \Rightarrow \quad f(4.12).f(4.13) < 0 \quad \Rightarrow \quad (-0.03).(0.05) < 0 \quad \textit{kök} \text{ var} \quad \boxed{\textit{kök} \Rightarrow x = 4.12}$$

Yer Değiştirme Yöntemi (Regula Falsi)

- En eski kök bulma yöntemlerinden birisidir.
- Eğrinin bir doğruyla yer değiştirmesi sonucunda, kökün konumunun yanlış belirlenmesi nedeniyle, latince "yanlış nokta" anlamında olan Regula Falsi olarak adlandırılır.
- Regula Falsi yönteminde köke yakınsama yavaş olmasına rağmen, mutlaka yakınsama vardır.
  - Bisection'dan hızlı, kiriş yönteminden yavaş
- f(x) fonksiyonunun [a, b] aralığında kökü hesaplanmak istensin
  - [a, f(a)] ve [b, f(b)] noktaları arasına bir kiriş (doğru) çizilir.
  - Doğrunun x eksenini kestiği noktanın (a_1) alt ve üst kısmında iki benzer üçgen oluşur.
  - İki üçgenin benzerliğinden x eksenini kestiği nokta (a_1) hesaplanır.
  - İstenilen hassasiyet (hata sınırı) sağlanmadıysa yukarıdaki işlemler [a_1 , f(a_1)] ve [b, f(b)] noktaları için tekrar ettirilir.



Yer Değiştirme Yöntemi (Regula Falsi)



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

$$\boxed{x_r = x_{ü} - \frac{f(x_{ü})(x_a - x_{ü})}{f(x_a) - f(x_{ü})}}$$

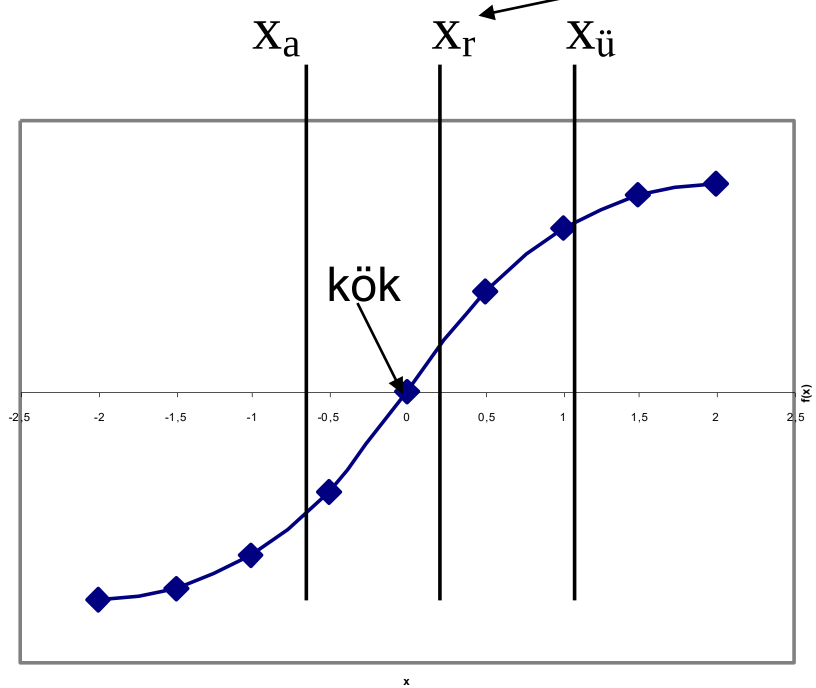
Serhat Yılmaz'ın Sunusundan Alınmıştır.

Yer Değiştirme Yöntemi (Regula Falsi)

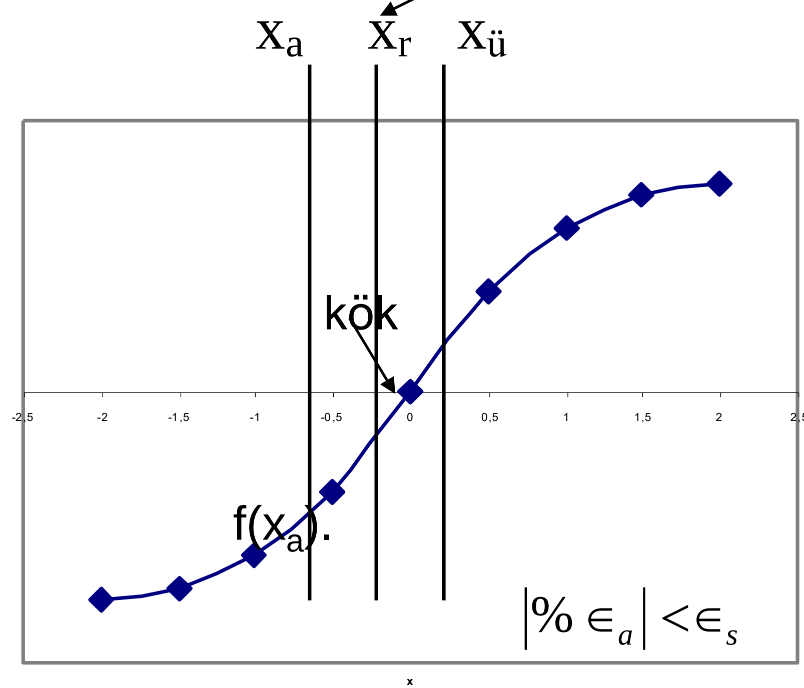
$$x_r = x_ü - \frac{f(x_ü)(x_a - x_ü)}{f(x_a) - f(x_ü)}$$

Güncellenecek sınır

- $f(x_a).f(x_r) < 0$ x_a ile x_r farklı bölgelerde xü(yeni)=xr
- $f(x_a).f(x_r) > 0$ x_a ile x_r aynı bölgelerde xa(yeni)=xr



Kök, x_a, x_r arasında



Kök, x_r, $x_ü$ arasında

Yer Değiştirme Yöntemi (Regula Falsi)

❖ **Örnek : Kütlesi m=68.1kg olan bir paraşütçünün, t=10 s serbest düştükten sonra 40m/s hıza sahip olabilmesi için gerekli direnç katsayısını yer değiştirme yöntemiyle iki iterasyon adımı için belirleyin.g:yer çek. =~9,8 kgm/s^2 (x_a=12, $x_ü$=16)**

$$f(c)=\frac{g\,m}{c}\left(1-e^{-(c/m)t}\right)-v$$

Çözüm: Burada kök x=c direncidir,

❑ 1. iterasyon:

x_a=12 ⟶ $f(x_a)$=6.0699

$x_ü$=16 ⟶ $f(x_ü)$=-2.2688

$$x_r=16-\frac{-2.2688(12-16)}{6.0669-(-2.2688)}=14.9113$$

$f(x_r)$=-0.25413

❑ 2. iterasyon: $f(x_a)*f(x_r)$= -1.5426 <0

x_r , $x_ü$ ile aynı bölgede olduğu için bir sonraki iterasyonun üst sınırı olacaktır.

$x_ü$=14.9113 ⟶ $f(x_ü)$= -0.2543

x_a=12 ⟶ $f(x_a)$=6.0699

$$x_r=14.9113-\frac{-0.2543(12-14.9113)}{6.0669-(-0.2543)}=14.7942$$

Serhat Yılmaz’ın Sunusundan Alınmıştır.

Yer Değiştirme (Regula Falsi) Yöntemi

❖ **Örnek :** **f(x) = x² – 49** denkleminin gerçek kökünün 7 olduğu bilindiğine göre **[5 , 9]** aralığındaki kök değerini **yer değiştirme** (regula falsi) **metodunu** kullanarak mutlak hata yüzdesi % ε_s =0.5 in altına ininceye kadar yaklaşık olarak bulunuz?

Not: Tüm değerler virgülden sonra 4 basamak alınacak.

$$x_r = x_{ü} - \frac{f(x_{ü})(x_a - x_{ü})}{f(x_a) - f(x_{ü})}$$

❶ İterasyon

$$x_r = 9 - \frac{32(5-9)}{-24-32} = 9 - \frac{-128}{-56} = 9 - 2,2857 = 6,7143$$

$$\%\varepsilon = |gercek - yaklasik| * 100 = |7 - 6.7143| * 100 = \%28.57$$

Kök hangi kısımda?

$$f(x_r).f(x_{ü}) < 0 \quad \Rightarrow \quad x_a = x_r$$

❷ İterasyon

$$x_r = 9 - \frac{32(6.7143-9)}{-3.9182-32} = 9 - \frac{-73.1424}{-35.9182} = 9 - 2,0363 = 6,9637$$

$$\%\varepsilon = |7 - 6.9637| = \%3.63$$

Kök hangi kısımda?

$$f(x_r).f(x_{ü}) < 0 \quad \Rightarrow \quad x_a = x_r$$

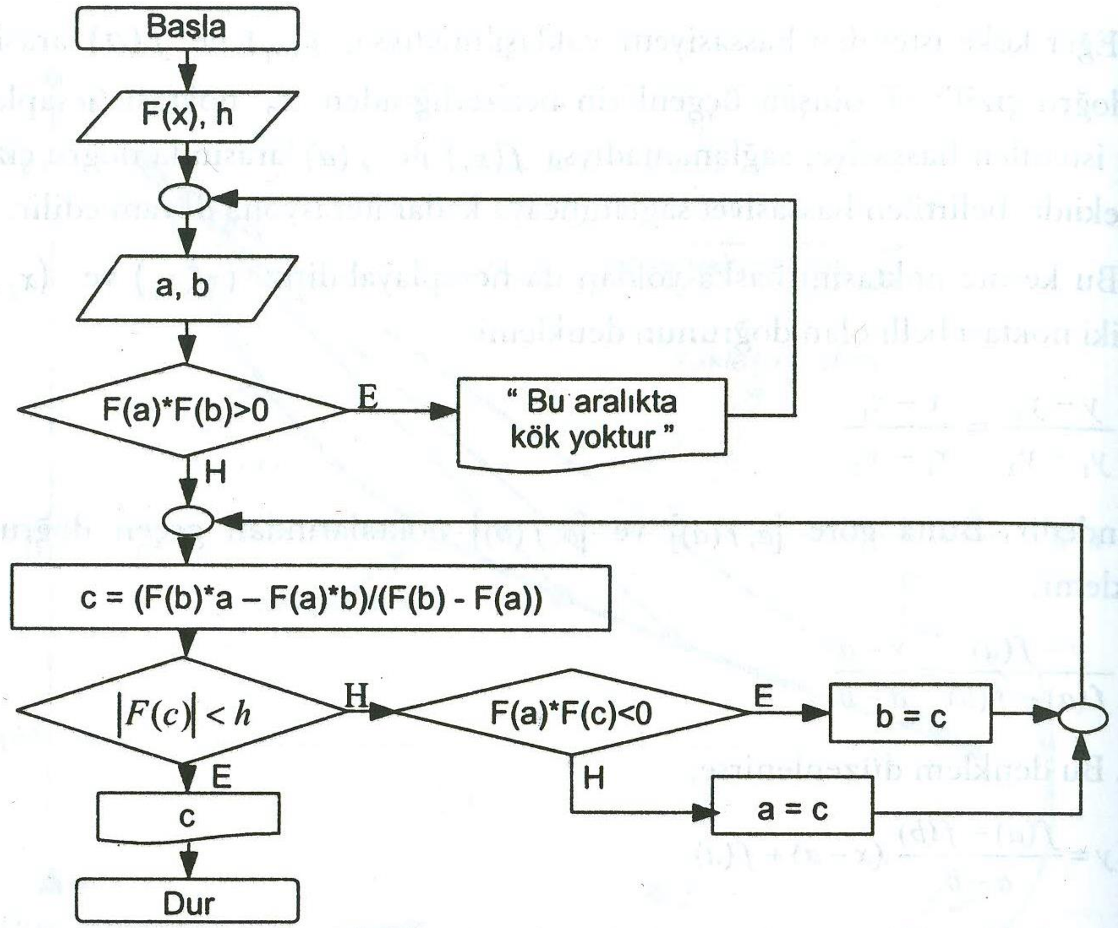
❸ İterasyon

$$x_r = 9 - \frac{32(6.9637-9)}{-0.5069-32} = 9 - \frac{-65.1616}{-32.5069} = 9 - 2,0045 = 6,9955$$

$$\%\varepsilon = |7 - 6.9955| = \%0.45$$

Regula Falsi Yöntemi

- **Algoritması**



Yer Değiştirme (Regula Falsi) Yöntemi(Ödev)

- **Örnek :** **f(x) = e^x – 2cos(x)** denkleminin $\left[0, \frac{\pi}{2}\right]$ aralığında köküne **Regula Falsi yöntemi** ile 3 iterasyon yaklaşınız? Virgülden sonra 4 basamak alınız.

Soru

- **f(x) = 3x + sin(x) - e**$^{-x}$ fonksiyonunun [**0 , 0.5**] aralığında kökünün olup olmadığını kontrol edip, eğer var ise kök değerini **yarılama** (bisection) **metodunu** kullanarak $\varepsilon_s = 0.05$ in altına ininceye kadar yaklaşık olarak bulunuz?

Not: Tüm değerler virgülden sonra 4 basamak alınacak.

KAYNAKLAR

- Serhat YILMAZ, "*Bilgisayar İle Sayısal Çözümleme*", Kocaeli Üniv. Yayınları, No:168, Kocaeli, 2005.
- Steven C. Chapra, Raymond P. Canale (Çev. H. Heperkan ve U. Kesgin), "*Yazılım ve Programlama Uygulamalarıyla Mühendisler İçin Sayısal Yöntemler*", Literatür Yayıncılık.
- İlyas ÇANKAYA, Devrim AKGÜN, Sezgin KAÇAR "*Mühendislik Uygulamaları İçin MATLAB*", Seçkin Yayıncılık
- Yüksel YURTAY, Sayısal Analiz Ders Notları, Sakarya Üniversitesi
- Prof.Dr. Asaf Varol, Sayısal Analiz Ders Notları, Fırat Üniversitesi