

BIL301 SAYISAL YÖNTEMLER

3. Hafta

Sayısal Yöntemlerde Hatalar

Doç. Dr. Sercan YALÇIN

Sayısal Hesaplamalardaki Hatalar, Hata Kaynakları

1. Modelleme:

2. Matematiksel Yöntemin Belirlenmesi:

3. Programlama:

4. Programın Çalıştırılması:

5. Sonuçların Yorumlanması:

2. adımda belirtildiği gibi, en az hatayı oluşturacak yöntemin seçilmesi ve 5. adımdaki hata toleransı, sayısal yöntemlerin etkili bir biçimde kullanılabilmesi için önemlidir.

Günlük Hayatımızda, Okulda, Mühendislik Hayatımızda Hataların Bedelleri

Bir problemin sayısal çözümlemesi yapılırken, bilgisayarlar sonlu sayıda rakamı saklayabilirler. Bu nedenle hesaplamalar tam değil yaklaşımlarla yapılabilir.

Burada önemli olan soru şudur: Hesaplarda ne kadar hata vardır ve bu kabul edilebilir mi?

Bu bölümde çeşitli hata tanımları, hataların tespiti ve en aza indirilmesi ile ilgili temel konular ele alınacaktır.

•3.1. Hata Tanımları

- Sayısal hatalar, gerçek matematik işlemlerinin ve büyüklüklerinin yaklaşımlarla ifade edilmelerinden doğar.

3.1. 1. Hata:

$$\begin{array}{c} \sim \\ a \\ a \end{array} \longrightarrow \epsilon = a - \begin{array}{c} \sim \\ a \end{array}$$

•3.1. Hata Tanımları

3.1. 1. Bağıl Hata: ϵ_r

$$\epsilon_r = \frac{\epsilon}{a} = \frac{a - \tilde{a}}{a} = \frac{\text{Hata}}{\text{Gerçek Değer}} \quad (a \neq 0)$$

a, tam olarak bulunamayacağı için bu formül kullanışsızdır

1. $|\epsilon|$, $\left| \frac{\tilde{a}}{a} \right|$ 'ya göre çok daha küçük ise a yerine \tilde{a} kullanılabilir. Bu durumda;

•3.1. Hata Tanımları

3.1. 1. Bağıl Hata: ϵ_r

$$\epsilon_r = \frac{\epsilon}{a} = \frac{a - \tilde{a}}{a} = \frac{\text{Hata}}{\text{Gerçek Değer}} \quad (a \neq 0)$$

$$|\epsilon| \leq \beta, \quad |a - \tilde{a}| \leq \beta$$

ölçebildiğimiz en büyük duyarlılıkta bir hata sınırı tanımlarız.

2.

•3.1. Hata Tanımları

3.1.3. Yaklaşım Hatası:

$$\epsilon_a = \frac{a_1 - a_0}{a_1}$$

•3.1. Hata Tanımları

3.1.4. Mutlak Hata

$$|\epsilon_a| \leq \epsilon_s$$

3.1.5. Yüzde hata: $\% 100 * \text{Hata}$

$$0.02 = 2/100 = \%2$$

- **Soru:** Bir köprü ve bir perçinin uzunluklarını ölçmeniz isteniyor. Ölçüm sonucunda boylarını sırasıyla 9999 ve 9 cm bulduğunuzu varsayalım. Eğer gerçek değerler sırasıyla 10 000 ve 10 cm ise her iki durum için (a) gerçek hatayı, (b) gerçek yüzde hatayı hesaplayın.

Çözüm:

Köprünün ölçülmesindeki hata

$$\epsilon_{\text{kopru}} = a_{\text{kopru}} - \tilde{a}_{\text{kopru}} = 10\,000 - 9999 = 1 \text{ cm}$$

perçinin ölçülmesindeki hata;

$$\epsilon_{\text{percin}} = 10 - 9 = 1 \text{ cm.}$$

Köprü için gerçek bağıl yüzde hata;

$$\% \epsilon_r (\text{kopru}) = \frac{\epsilon_{\text{kopru}}}{a_{\text{kopru}}} * \% 100 = \frac{1}{10000} * \% 100 = \% 0.01$$

Perçin için gerçek bağıl yüzde hata;

$$\% \epsilon_r (\text{percin}) = \frac{\epsilon_{\text{percin}}}{a_{\text{percin}}} * \% 100 = \frac{1}{10} * \% 100 = \% 10$$

Hataları anlamlı basamak sayısıyla ilişkilendirmek
(Scarborough, 1966)

$$\pi = 3.14159265358979 \quad \tilde{\pi} = 3.14639184292698$$

$$\pi = 3.\underbrace{14159265358979}_n$$

$$\epsilon_{\pi} = \pi - \tilde{\pi}$$

$$\epsilon_s = \% \left(0.5 * 10^{2-n} \right)$$

Örnek: Matematikte işlevler çoğu zaman sonsuz serilerle gösterilebilir. Örneğin , üstel işlev aşağıdaki açılım kullanılarak hesaplanabilir.

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots + \frac{x^n}{n!}$$

Böylece daha fazla sayıda terim eklendikçe, yaklaştırma, 'in gerçek değerini daha iyi tahmin eder (Mac Laurin Serisi Açılımı).

Soru: İlk terimden başlayarak ($e = 1$), tek tek terimler ekleyerek e sayısını tahmin edin x
(Gerçek değer $e=1.648721\dots$) .

Her adımda gerçek ve yüzde bağıl hatalarını bulun. 0.5

Yaklaşık hata 'nın mutlak değeri , üç anlamlı basamak veren belirli bir hata kriterinden daha küçük oluncaya kadar terim eklemeye devam edin .

Çözüm: Öncelikle sonucun en az üç anlamlı basamak için doğru olmasını garanti eden hata kriteri belirlenebilir.

$$\epsilon_s = \% \left(0.5 * 10^{2-3} \right) = \% 0.05$$

Bu seviyenin altına inene kadar terim eklenmeye devam edilir

Seriye bakarsak eklenecek ilk terim 1'dir. Bu nedenle $e^{0.5}$ değeriyle ilgili ilk tahminimiz;

$$e^x = e^{0.5} = 1 \quad \% \epsilon_r = \frac{1,648721 - 1}{1,648721} * \% 100 = \% 39.34$$

2. tahmin değeri ikinci terim eklenerek bulunur

$e^x = 1+x$ ve $x=0.5$ için $e^{0.5} = 1+0.5$ Bu değerden , gerçek bağıl yüzde hata bulunur.

$$\% \epsilon_r = \frac{1,648721 - 1.5}{1,648721} * \% 100 = \% 9.02$$

Bağıl yüzde hatanın yaklaşık bir tahmini olan yüzde yaklaşım hatası ise şu şekildedir:
 1. adımda elimizdeki tek tahmin değeri 1 olduğu için bir hata karşılaştırması yapamayız. Şayet önceki tahmin değerini 0, mevcut tahmin değerini 1 seçersek hata %100 çıkar. 2. adımda

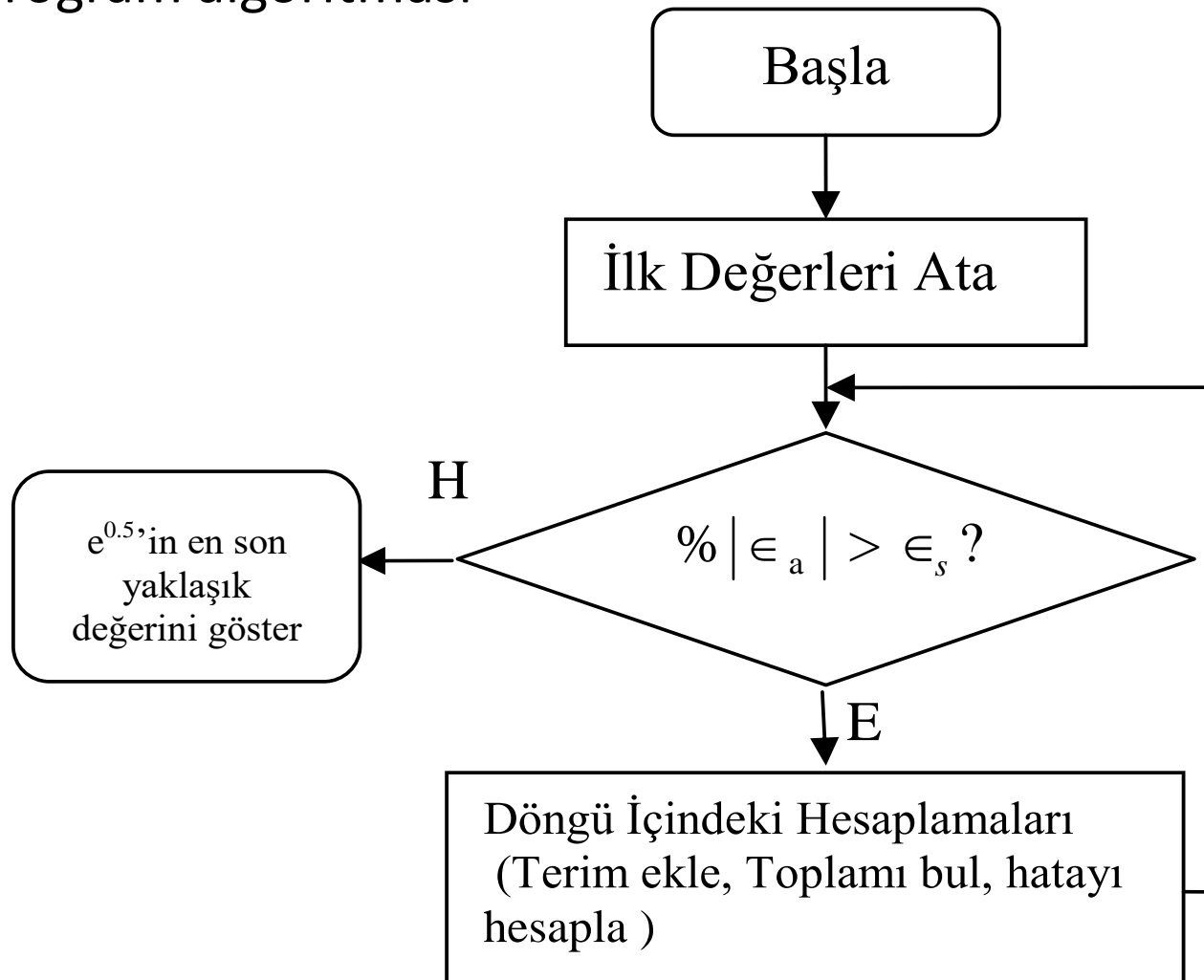
$$\% \epsilon_a = \frac{1.5 - 1}{1.5} * \% 100 = \% 33.3 \quad \epsilon_a > \epsilon_s$$

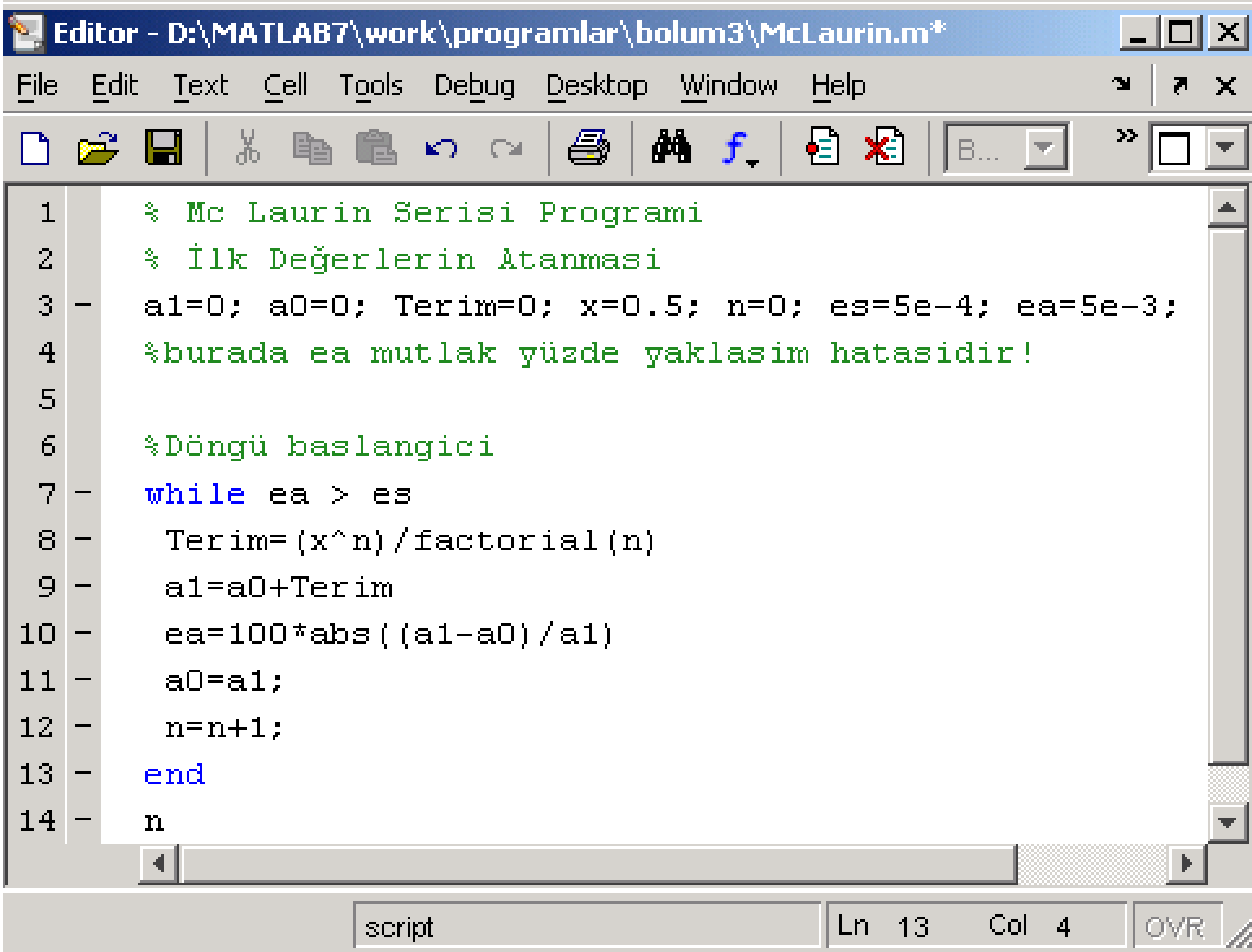
Eklenecek terim $\frac{x^2}{2!}$

$$e^{0.5} = 1 + 0.5 + \frac{0.5^2}{2} = 1.625$$

İşleme $\% \epsilon_a < \epsilon_s$ oluncaya kadar devam edilir.

- Problemi Mc Laurin serisi yöntemine göre çözecek olan program algoritması





The image shows a MATLAB Editor window titled "Editor - D:\MATLAB7\work\programlar\bolum3\McLaurin.m*". The window has a menu bar with "File", "Edit", "Text", "Cell", "Tools", "Debug", "Desktop", "Window", and "Help". Below the menu bar is a toolbar with various icons for file operations, editing, and execution. The main editing area contains a script with the following code:

```
1      % Mc Laurin Serisi Programi
2      % İlk Değerlerin Atanması
3      - a1=0; a0=0; Terim=0; x=0.5; n=0; es=5e-4; ea=5e-3;
4      %burada ea mutlak yüzde yaklasim hatasidir!
5
6      %Döngü baslangici
7      - while ea > es
8      -     Terim=(x^n)/factorial(n)
9      -     a1=a0+Terim
10     -     ea=100*abs((a1-a0)/a1)
11     -     a0=a1;
12     -     n=n+1;
13     - end
14     - n
```

The status bar at the bottom shows "script", "Ln 13", "Col 4", and "OVR".

3.2. Kesme ve Yuvarlama Hataları:

- Kesme
 - 1.59, 1.5 olarak algılanır. Hata, 0.09'dur
 - 1.51, 1.5 olarak algılanır. Hata 0.01'dir
- Yuvarlama
 - 1.59, 1.6 olarak algılanır. Hata, 0.01'dir.
 - 1.51, 1.5 olarak algılanır. Hata 0.01'dir.

Sayıların Kayan Noktalı Olarak Gösterilimi (Floating Point):

- Kesirli nicelikler ;

$$m * b^e$$

Tablo.3.1. Kayan noktalı gösterilim

İşaret	İşareti, Üs	Mantis
--------	-------------	--------

Örneğin bir aktarma işlemi sonucunda, 0.029411765 sayısını sadece 4 ondalık basamağının saklandığını varsayalım. Bu sayı normalde olduğu gibi alınırsa ondalık tabanda şu şekilde saklanır.

0.0294*10⁰ kayan noktayla 0.2941*10⁻¹

Örnek-1 Körfez savaşı sırasında 25 Şubat 1991 günü



atılan Irak Skud füzesi



Suudi Arabistan'daki
Amerikan patriot füzesi

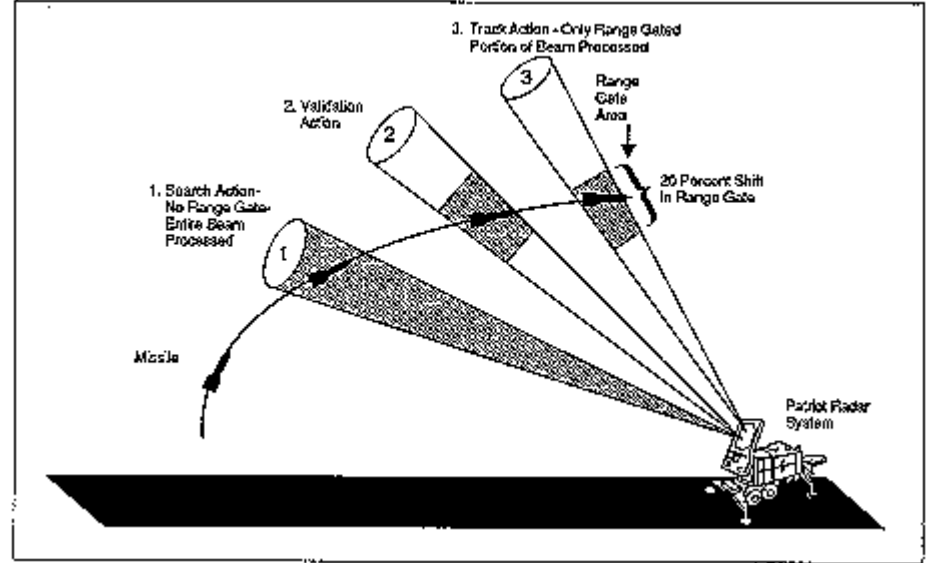


- algılamadı ve 28 kişi öldü.

Soruşturma sonunda, sorunun bir yazılım hatasından kaynaklandığı anlaşıldı.

- Patriot bilgisayar aritmetik hata nedeniyle füze zamanını yanlış hesaplamıştı.

Figure 4: Calculated Range Gate After Approximately 8 Hours



Bilgisayar saatinden alınan değer 1/10 ile çarpılarak geçen süre hesaplanmaktaydı. Bilgisayar 24-bit sabit kayıtcı kullanmaktaydı.

$1/10$ sayısı 2 'li tabanda $1/2 + 1/2 + 1/2 + 1/2 + 1/2 + 1/2 + \dots$ şeklinde sonsuz terimlidir. Başka bir deyişle $1/10$ 'un 2 li tabandaki açılımı $0.0001100110011001100110011001100\dots$ şeklindedir.

- [illegible]

100 saatlik batarya süresince bu hata:

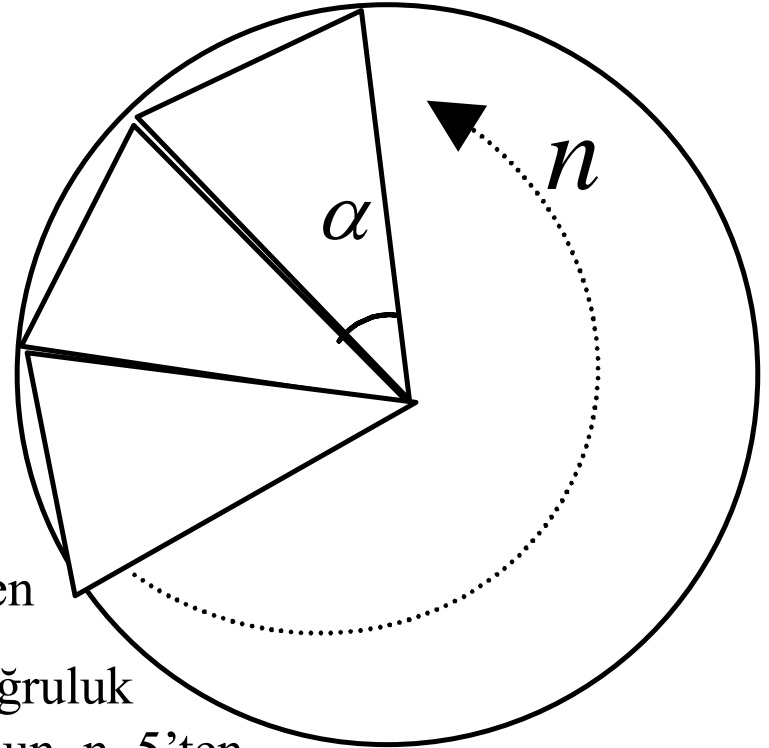
$$0.0000000095 \times 100 \times 60 \times 60 \times 10 = 0.34 \text{ sn}$$

Saniyede 1,676 metre yol alan skud füzesi 0.34 saniyede 500 metreden fazla gider.

Bu kadarlık hata ise Skud'un menzil dışında görülmesine yol açar (Sevgi, L.,2005).

Örnek-2: Bir matematikçi yandaki şeklin alanını analitik olarak $A_D = \pi r^2$ bulmuş ve bu şekle daire adını vermiştir.

- Sadece ikiz kenar üçgenlerin alanını hesaplayabilen bir bilgisayarla bu dairenin alanını % $\epsilon_a < 0.01$ hata ile bulabilecek bir algoritma oluşturun ve programını Matlab programlama dilinde yazın.

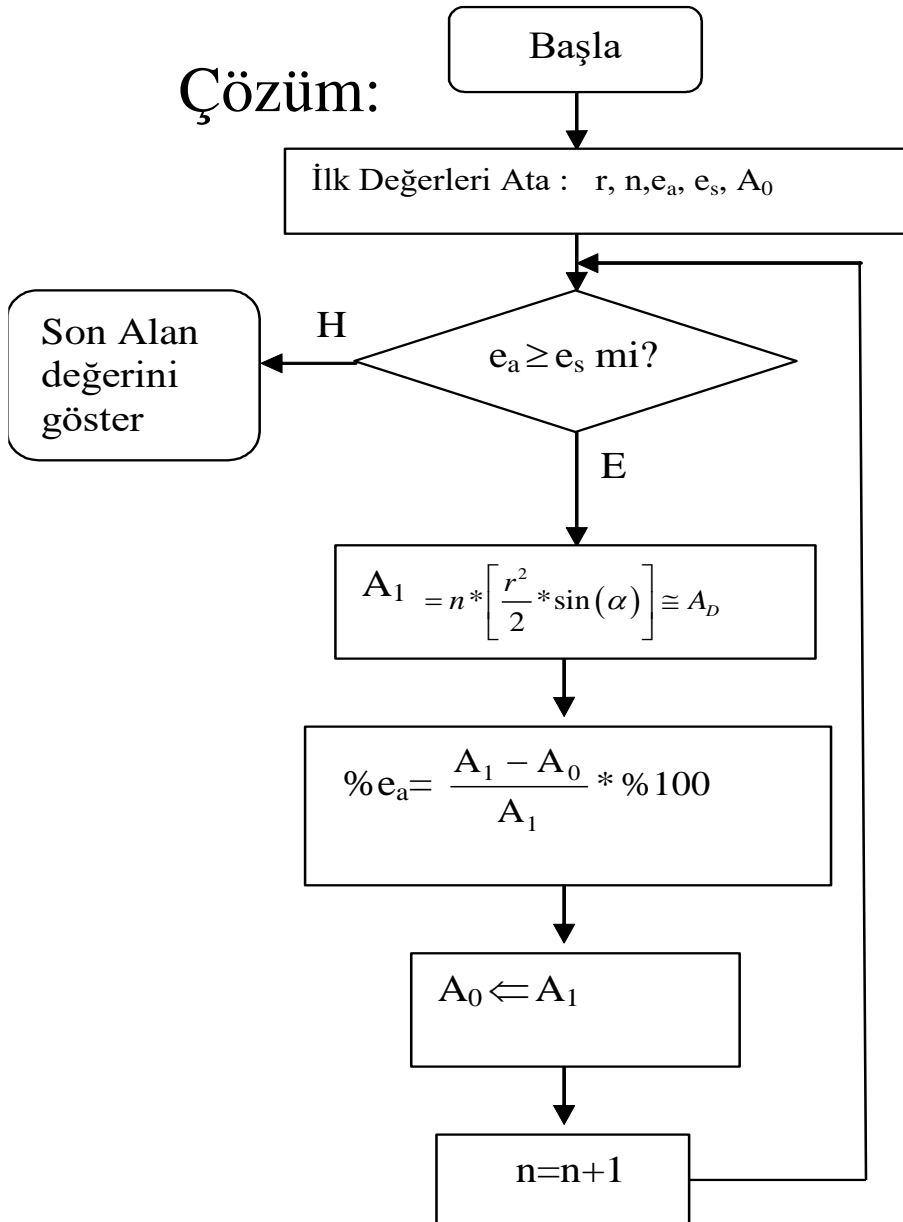


Program, her adımda daireyi üçgenlerden oluşan n eşit parçaya bölsün, istenen doğruluk sağlanmadıkça n 'i 1 arttırsın ($r=1$ cm olsun, n , 5'ten başlasın). Daireyi yaklaşık olarak oluşturacak n tane üçgenin alanı ;

$$n * A_{\text{üçgen}} = n * \left[\frac{r^2}{2} * \sin(\alpha) \right] \cong A_D$$

$$\alpha = \frac{2\pi}{n}$$

Çözüm:



```

Editor - D:\MATLAB7\work\programlar\bo...
File Edit Text Cell Tools Debug Desktop Window Help
1 - r=1; n=5; ea=0.1; es=0.01;
2 - A0=0;
3 - while ea>=es
4 -     A1=n*(r^2/2*sin(2*pi/n));
5 -     ea=abs((A1-A0)/A1)*100
6 -     A0=A1;
7 -     n=n+1
8 - end
9
10 - A1
  
```

The MATLAB Editor window shows the code implementation of the algorithm. The code initializes $r=1$, $n=5$, $ea=0.1$, $es=0.01$, and $A0=0$. It enters a `while` loop that continues as long as $ea \geq es$. Inside the loop, it calculates $A1 = n * (r^2 / 2 * \sin(2 * \pi / n))$, updates the error $ea = \text{abs}((A1 - A0) / A1) * 100$, updates $A0 = A1$, and increments n by 1. The loop ends with the `end` statement. The current line of code is line 10, which is `A1`.