TANA21: Projektrapport

Integration

Namn	**Personnummer**	**Epostaddress**
Martin Söderén	900929-1098	marso329@student.liu.se
Alexander Yngve	930320-6651	aleyn573@student.liu.se

1 Inledning

Denna rapport behandlar hur noggrann man kan integrera med hjälp av trapetsregeln samt den aritmetiska komplexiteten för integreringen.

2 Uppgift

Projektets uppgift är att skapa en MATLAB-funktion trapezoid som integrerar en funktion numeriskt med hjälp av trapetsregeln. Som argument tar funktionen in en funktion, startpunkt, slutpunkt och antalet delintervall.

Frågor att besvara:

- Är noggrannhetsordningen som förväntad?
- Är den aritmetiska komplexiteten som förväntad?

3 Teori

Den matematiska definitionen ser ut som följande och är tagen från kursboken avsnitt 7.2.

$$\int_a^b f(x)dx = T(h) + R_T$$

$$T(h) = h(\frac{1}{2}f_0 + f_1 + ... + f_{n-1} + \frac{1}{2}f_n)$$

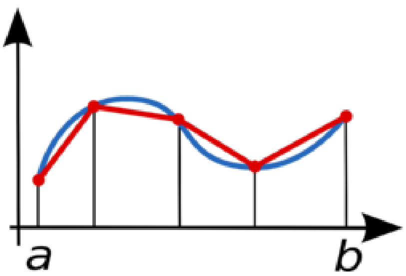
Trunkeringsfelet ges av

$$R_T = -\frac{b-a}{12}h^2 f''(n)$$

om f'' är kontinuerlig.

Definitionen av trapetsregeln från kursboken s.168

Det detta gör är en approximation av integralen genom att ställa upp en massa trapetser med bredd h under kurvan och summerar dessas areor. Se figur 1.



Figur 1 – Trapetsregeln

4 Metod

Funktionerna implementerades enligt metoden beskriven i avsnitt 7.2 i kursboken. Därefter testades *trapezoid* med koden i figur 3 och i figur 4. Det testkoden i figur 4 gör är att för de testfunktionera som var givna i projektbeskrivningen integerera dessa med hjälp av *trapezoid* och använda det analytiskt framtagna värden för integrationen och få fram felet. Dessa fel logaritmeras sedan och lutningen på kurvan används för att avgöra hur exakt trapetsmetoden är på funktionen.

Därefter testas tidskomplexiteten genom att uföra integereringar på e^x med olika h och mäta tiden det tar.

5 Kod

Funktionen *trapezoid* (figur 2) integrerar funktion f mellan a och b med hjälp av n stycken delintervall.

```
function I =  trapezoid(f,a,b,n)
h=(b-a)/n;
I=(f(a)+f(b))/2;
for k=1:n-1
    I=I+f(a+k*h);
end
I=h*I;
end
```

Figur 2 – *trapezoid*

```
%test time complexity
n=[1 2 4 8 16 32 64 128 256 512 1024 2048 4096];
times=[];
for x=n
    tic;
trapets=trapezoid(@exp,0,1,x);
times=[times,toc];
end
plot(n,times);
xlabel('Intervals')
ylabel('Time')
```

Figur 3 – Testkod för aritmetisk komplexitet

```
n=[1 2 4 8 16 32 64 128 256 512 1024 2048 4096];
data=zeros(7,length(n));
i=1;
for x=n
    %e^x [0 1]
    data(1,i)=abs((exp(1)-exp(0))-trapezoid(@exp,0,1,n(i)));
```

```
    %3*x^2+2*x+1 [0 1]
    f=@(x) 3*x^2+2*x+1;
    f_int=@(x) x^3+x^2+x;
    data(2,i)=abs((f_int(1)-f_int(0))-trapezoid(f,0,1,n(i)));

    %2*x+2 [0 1]
    f=@(x) 2*x+2;
    f_int=@(x) x^2+2*x;
    data(3,i)=abs((f_int(1)-f_int(0))-trapezoid(f,0,1,n(i)));

    %5*x^4+3*x^3+4*x^2+x+15 [0 1]
    f=@(x) 5*x^4+3*x^3+4*x^2+x+15;
    f_int=@(x) x^5+(3/4)*x^4+(4/3)*x^3+(1/2)*x^2+15*x;
    data(4,i)=abs((f_int(1)-f_int(0))-trapezoid(f,0,1,n(i)));

    %4/(1+x^2) [0 1]
    f=@(x) 4/(1+x^2);
    data(5,i)=(4*(atan(1)-atan(0)))-trapezoid(f,0,1,n(i));

    %x^(1/2) [0 1]
    f=@(x) x^(1/2);
    f_int=@(x) (2/3)*x^(3/2);
    data(6,i)=abs((f_int(1)-f_int(0))-trapezoid(f,0,1,n(i)));

    %sin(x) [0 2*pi]
    f=@(x) sin(x);
    f_int=@(x) -cos(x);
    data(7,i)=abs((f_int(2*pi)-f_int(0))-trapezoid(f,0,2*pi,n(i)));

    i=i+1;
end
clf
hold on;
for j=[1:1:6]
plot(log(n),log(data(j,:)))
log_data=log(data(j,:));
log_axis=log(n);
(log_data(length(data(j,:)))-log_data(1))/(log_axis(length(log_axis))-log_axis(1))
end
xlabel('log(h)')
ylabel('log(error)')
legend('e^x','3*x^2+2*x+1', '2*x+2','5*x^4+3*x^3+4*x^2+x+15','4/(1+x^2)','x^{1/2}')

data;
```

Figur 4 – Testkod för felen

6 Validering

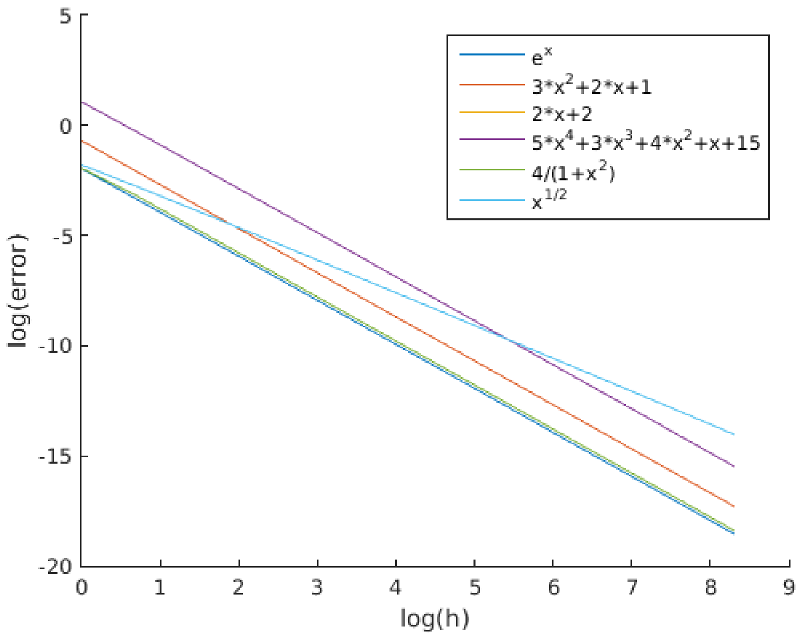
För samtliga funktioner i testkoden så togs en analytiskt integration fram som användes till referens vid testen. Inget fel var större än 10^{-5} vilket gör att man kan anta att funktionen räknar rätt. Samliga fel för alla funktioner återfinns under resultat.

7 Resultat

Resultatet av testet beskrivet i avsnitt 4 återges i tabell 1.

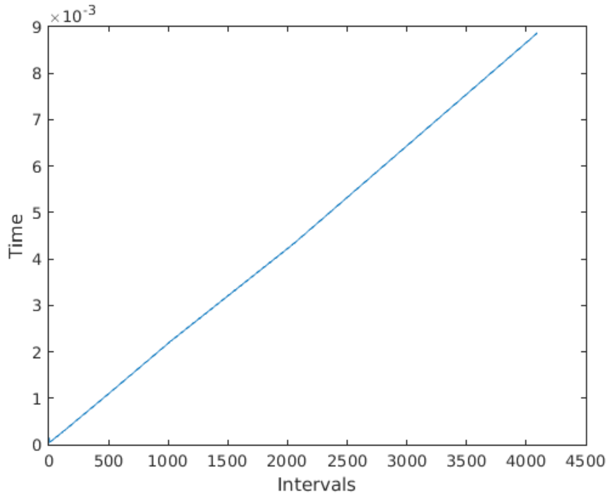
Funktion [gränser]	**Noggrannhetsordning**
e^x [0 1]	2
$3x^2 + 2x + 1$ [0 1]	2
$2x + 2$ [0 1]	0
$5x^4 + 3x^3 + 4x^2 + x + 15$ [0 1]	2
$4/(1 + x^2)$ [0 1]	2
$x^{(1/2)}$ [0 1]	1.5
$sin(x)$ [0 2π]	0

Tabell 1 – Resultat för de olika funktionernas noggrannhetsordning



Figur 5 – Felet vid olika antal intervall.

Från figur 5 kan man urskilja att felet avtar snabbt då h ökar. Resultatet i tabell 1 togs fram genom lutningen på varje funktions linje i figur 5. $sin(x)$ är inte med i figuren då dess fel var ≈ 0 vilket gav en konstig linje. Anledningen till att $sin(x)$:s fel var noll beror på att den är periodisk. $2x + 2$:s fel var också noll och detta beror på att dess andraderivata är noll och att dess integral blir väldigt exakt om approximerar den med trapetsmetoden.



Figur 6 – Tiden vid olika antal intervall.

Från figur 6 kan man se att tidskomplexiteten är linjärt beroende av antalet intervall man använder eller längden h på alla trapetser. Detta är rimligt med hänsyn på definitionen då ett extra intervall ger en extra term att addera.

8 Diskussion

Noggranhetsordningen är som förväntat på två. Det vill säga den dominerande feltermen är proportionell mot h^2. Den aritmetiska tidskomplexiteten är linjär mot h det vill säga den är $\mathcal{O}(n)$, vilket är som förväntat.