Mathematics for Robotics Assignment 3

Root Solving, Numerical Differentiation, Numerical Integration, Ordinary Differential Equations

Per Henrik Hardeberg
se23mrob005

University of Agder - 2023
Faculty of Engineering and Science

Contents

List of Figures

1 Task 1 Find Root

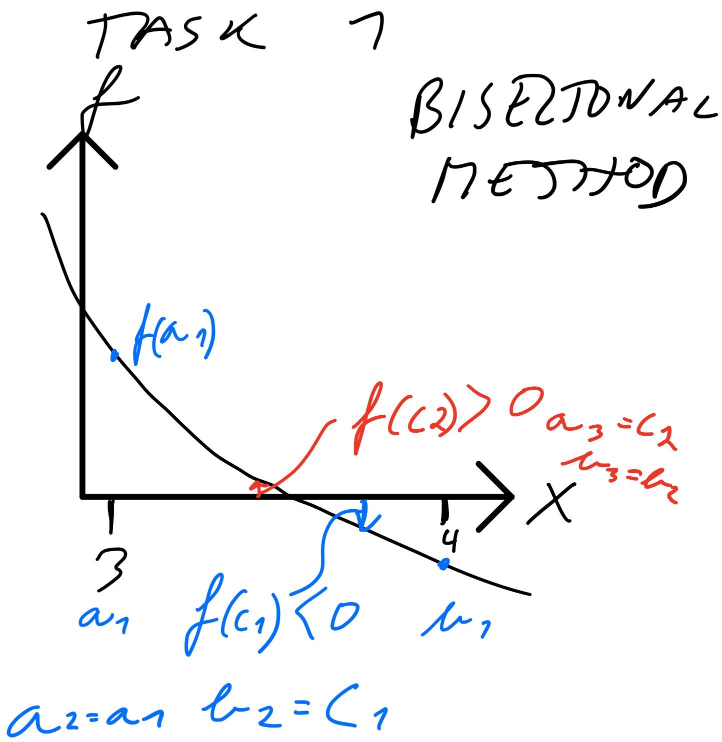


Figure 1: Task 1

```
clear; clc; close all;
% Task 1 Using bisectional method

syms x
xList = 3 : 0.001 : 4;

f = exp(-x)*(3.2*sin(x) - 0.5*cos(x));
f_plot = double(subs(f,x,xList));
a1 = 3;
b1 = 4;
c1 = (b1+a1)/2;
f_c1 = double(subs(f,x,c1))

% f_c1 < 0, therefore
a2 = a1;
b2 = c1;
c2 =(b2+a2)/2;
f_c2 = double(subs(f,x,c2))

% f_c2 > 0, therefore
a3 = c2;
b3 = b2;
c3 =(b3+a3)/2;
f_c3 = double(subs(f,x,c3))

% f_c3 < 0, therefore
a4 = a3;
b4 = c3;
c4 =(b4+a4)/2;
f_c4 = double(subs(f,x,c4))


```

```
% f_c4 < 0, therefore
a5 = a4;
b5 = c4;
c5 =(b5+a5)/2;
f_c5 = double(subs(f,x,c5))

%

% f_c5 > 0, therefore
a6 = c5;
b6 = b5;
c6 =(b6+a6)/2;
f_c6 = double(subs(f,x,c6))

plot(xList,f_plot)
hold on
plot(c1,f_c1, '*')
plot(c2,f_c2, '*')
plot(c3,f_c3, '*')
plot(c4,f_c4, '*')
plot(c5,f_c5, '*')
plot(c6,f_c6, '*')
grid on
xlim([3.2 3.6])
legend('f(x)', ...
    '(c2,f(c2))','(c3,f(c3))','(c3,f(c3))','(c4,f(c4))','(c5,f(c5))','(c6,f(c6))')
%Plottet to .eps in differnt script
```

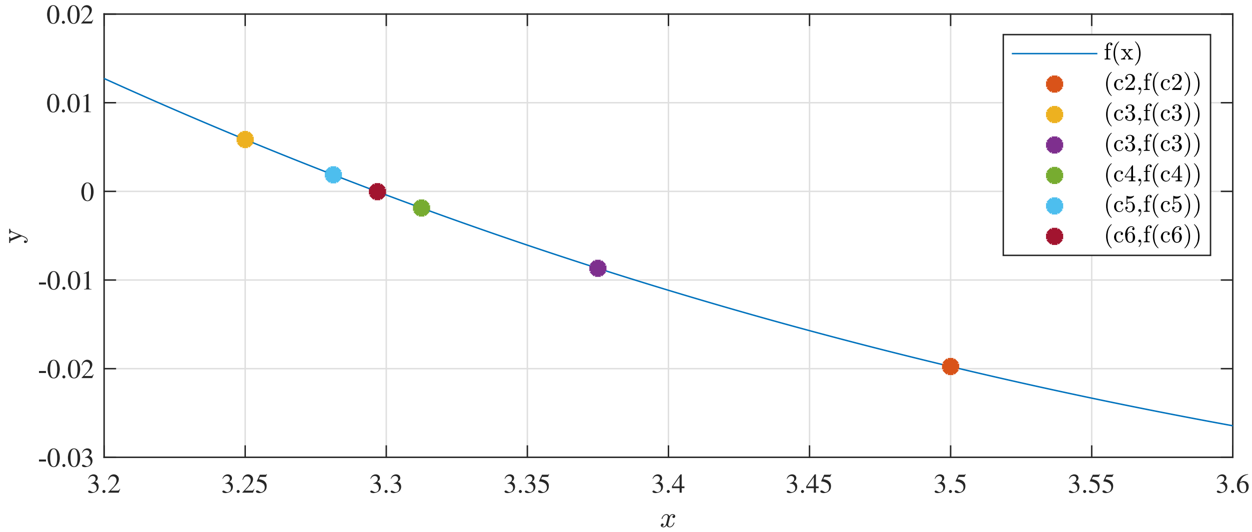
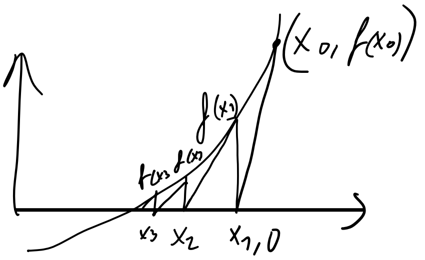


Figure 2: Plot 1

Root:

$$x = 3.2969$$

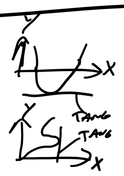
2 Task 2 Newton Raphson



LIMITATIONS:

$f'(x_n) = 0$

$f'(x) = \infty$



COULD ALSO OSCILLATE BETWEEN TWO

$$X_{n+1} = X_n - \frac{f(x_n)}{f'(x_n)}$$

$$f_1(x) = 4 - 8y_2 + 4y_3 - 2y_2^3 = 0$$

$$f_2(x) = 1 - 4y_2 + 3y_3 + y_3^2$$

$$X_{n+1} = X_n - \frac{f(x_n)}{f'(x_n)} = X_n - J^{-1} F(x)$$

$$F_0 = \begin{bmatrix} 0,5 \\ 0,5 \end{bmatrix} \qquad F(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \end{bmatrix}$$

$$J = \begin{bmatrix} \frac{\partial}{\partial y_2} f_1(x) & \frac{\partial}{\partial y_3} f_1(x) \\ \frac{\partial}{\partial y_2} f_2(x) & \frac{\partial}{\partial y_3} f_2(x) \end{bmatrix}$$

$$J = \begin{bmatrix} -8 - 6y_2^2 & 4 \\ -4 & 3 + 2y_3 \end{bmatrix}$$

$$F_{n+1} = F_n - J^{-1} F(x)$$

$$= \begin{bmatrix} 0,5 \\ 0,5 \end{bmatrix} - J^{-1}$$

Figure 3: Task 2

```
%% Task 2
clear; clc; close all;

syms y2 y3
f1 = 4 - 8*y2 +4*y3 - 2*y2^3;
f2 = 1 - 4*y2 + 3*y3 + y3^2;
F = [f1;f2];

%enteties of  jacobian matrix
Ja = diff(f1,y2);
Jb = diff(f1,y3);
Jc = diff(f2,y2);
Jd = diff(f2,y3);

J = [Ja, Jb;
    Jc, Jd];
init = [0.5; 0.5];
Values(:,1) = init;
J1 = subs(J,[y2,y3],init')

for i = 1:6
Jinv = double(subs(J,[y2,y3],Values(:,i)')^-1)
Fn = double(subs(F,[y2,y3], Values(:, i)'))
Values(:,i+1) = Values(:,i) - (Jinv*Fn)
i = i +1;
end

% Check values for y2 and y3
ans = double(subs(F,[y2,y3], Values(:, i)'))
```

Sixt itteration gives value of:

$$y1 = 0.6652$$
$$y2 = 0.4776$$

When these values are put back in the original function f1(x) and f2(x) answer is:

$$f1 = 0.2034 \cdot 1.0e - 15 \approx 0$$
$$f2 = 0.1405 \cdot 1.0e - 15 \approx 0$$

3 Task 3 Secant-Method

TASK 3

$f(x) = x^2 + 2\sin(x) + \cos(x)$

SECANT METHOD:

$$X_{n+1} = X_n - f(X_n)\left(\frac{X_i - X_{i-1}}{f(x_i) - f(x_{i-1})}\right)$$

USED TWO INIT CONDITIONS X_n AND X_{n-1}
(X_1) (X_0)

CHANGING TO

$X_1 = 0$ $X_2 = -0,1$

BECAUSE OF MATLAB

Figure 4: Task 3

```
%% Task 3
clear; close all; clc;
x(1) = 0;
x(2) = -0.1;
xList = -1 : 0.001: 0.1;
fPlot = xList.^2+2*sin(xList)+cos(xList);
for i = 2:6
fn = x(i)^2+2*sin(x(i))+cos(x(i));
fn_1 = x(i-1)^2+2*sin(x(i-1))+cos(x(i-1));
x(i+1)= x(i) - fn*(x(i)-x(i-1))/(fn-fn_1);
end

fPlottest = x.^2+2*sin(x)+cos(x);
plot(xList,fPlot)
hold on
plot(x,fPlottest, '*')
x(i)
%Plottet to .eps in differnt script
```

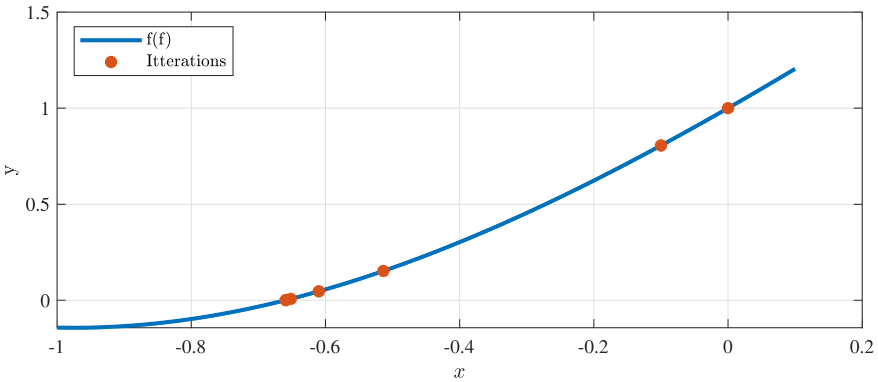


Figure 5: Plot 3

$$x_6 = -0.6588$$

4 Task 4 Numerical Differenciation

TASK 4

$h = 0{,}04$

$f(x) = \sin(x)$

$f'(x) = \cos(x)$

DIVIDED DIFF:

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h}$$

$f(x+h) = 0{,}9781$ $\quad f(x) = 0{,}9857$

$f(x-h) = 0{,}9916$

$f'(x) = -0{,}1688$

$$\left.\begin{aligned} y - y_1 &= a(x - x_1) \\ y &= f\_diff \cdot (x - 1{,}74) + 0{,}9857 \end{aligned}\right\}\text{PLOT TANGENT}$$

Figure 6: Task 4

```
%% task 4
clear; close all; clc;

x = 1.6 : 0.01 : 1.9;
y = sin(x);


hold on
xGiven = [1.7 1.74 1.78 1.82 1.86];
yGiven = [0.9916 0.9857 0.9781 0.9691 0.9584];


x_investigate = 1.74;
f_x_pluss_h = yGiven(3);
f_x_minus_h = yGiven(1);
h = 0.04;
f_diff_x = (f_x_pluss_h-f_x_minus_h)/(2*h)

tangent = f_diff_x.*(x-1.74)+0.9857;

plot(x,y)
plot(xGiven, yGiven, '*')
plot(x,tangent)
legend('sin(x)', 'Given points', 'Tangent')
```

I am using the divided difference formula. The tangent is plottet to see that the slope looks correct.

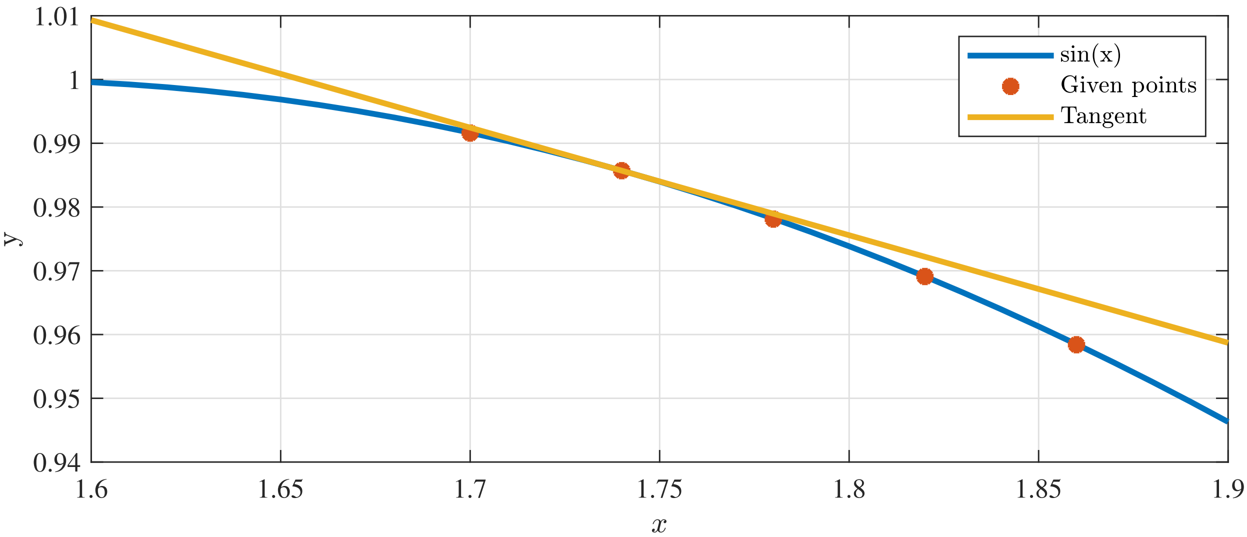


Figure 7: Plot 4

$$cos(1.74) \approx -0.1688$$

5 Task 5 Numerical Integration Simpsons 1/3

TASK 5

a) $$\int_a^b f(x)\,dx = \frac{h}{3}\Big((y_0+y_n)+4(y_1+y_3\dots)+2(y_2+y_4\dots)\Big)$$

$$h = \frac{b-a}{n} \qquad n=2$$

X_0 ✓ Y_0 ✓

X_1 ✓ Y_1 ✓

X_2 ✓ Y_2 ✓

$$\frac{h}{3}\Big((y_0+y_2)+4(y_1)\Big) = \int_8^{30} f(x)$$

b) ERROR FROM SIMPSONS 1/3

$$\int_a^b T_4\,dx = \frac{-h^5}{90} f^5(0) \qquad h^5$$

$$E_t = \frac{-h^5}{90} = -1,78\cdot 10^3$$

c) $$\frac{E_t}{SIMPS1\_3}\cdot 100 = -16\,\%$$

Figure 8: Task 5

```
%% task 5
clear; close all; clc;
% Nothing else given in task, so I choose n=2
n = 2;
a = 8;
b = 30;
h = (b-a)/n;

t = a; %x0
x0 = t;
```

```
y0 = 2000*log(140000/(140000-2100*t)) - 9.8*t
t = a+h; %x1
x1 = t;
y1 = 2000*log(140000/(140000-2100*t)) - 9.8*t
t = a+2*h; %x1
x2 = t;
y2 = 2000*log(140000/(140000-2100*t)) - 9.8*t

simps1_3 = h/3 * ((y0+y2)+4*(y1))
%b)
Et = -h^5/90
%c)
prosent = Et/simps1_3 * 100
```

$$x \approx 1.1066e + 04$$

6 Task 6 Numerical Integration Simpsons 3/8

```
%% task 6
clear; close all; clc;
% Choose n = 3
n = 3;
a = 8;
b = 30;
h = (b-a)/n;

t = a; %x0
x0 = t;
y0 = 2000*log(140000/(140000-2100*t)) - 9.8*t
t = a+h; %x1
x1 = t;
y1 = 2000*log(140000/(140000-2100*t)) - 9.8*t
t = a+2*h; %x1
x2 = t;
y2 = 2000*log(140000/(140000-2100*t)) - 9.8*t
t = a+3*h; %x1
x3 = t;
y3 = 2000*log(140000/(140000-2100*t)) - 9.8*t

simps3_8 = h*3/8 *(y0+3*y1+3*y2+y3)
% %b)
Et = -h^5*3/80
% %c)
prosent = Et/simps3_8 * 100
```

SEE MATLAB:

a) $n = 3$ - - -

b) $E = -795$

c) $= 7\%$ LESS BECAUSE $n = 3$ INSTED OF $n = 2$ IN LAST TASK

Figure 9: Task 6

$$x \approx 1.1063e + 04$$

From the error formla we can se that the error difference is small because both have h^5. However in this case $h = 2$ in task 5 and $h = 3$ in task 6. This makes the big difference in the error.

7 Task 7 Heun's Method

7

HEUNS METHOD:

$$y' = yx^2 - 1{,}2y$$

$$h = 0{,}5 \qquad X = [0, 1]$$

$$y(0) = 1 \qquad \boxed{\begin{matrix} x_0 = 0 \\ y_0 = 1 \end{matrix}}$$

$$k_1 = f(x_i, y_i)$$

$$k_2 = f(x_i + h, y_i + k_1 h)$$

$$y_{i+1} = y_i + (k_1 + k_2)\frac{h}{2}$$

CHANGE TO $X_0 \to X_1$ BECAUSE
$Y_0 \to Y_1$ OF MATLAB

Figure 10: Task 7

```
%% task 7
% dy_dt = f(x,y) = y*x^2 - 1.2*y = y(i)*x(i)^2 - 1.2*y(i)
clear; close all; clc;

%change itterations to get more accuracy
H = [0.5 0.001];
for j = 1:2
h = H(j);
itterations = 1/h;
%initial:
i = 1;
x(i) = 0;
y(i) = 1;

for i = 1 : itterations
x(i+1) = x(i) +h;
k1(i) = y(i)*x(i)^2 - 1.2*y(i);
k2(i) = (y(i)+k1(i)*h)*(x(i+1)) ^2 - 1.2*(y(i)+k1(i)*h);
y(i+1) = y(i) + (k1(i)+k2(i))*h/2;
end
```

```
plot(x,y)
hold on
end

legend('Heuns Method h = 0.5', 'Heuns Method h = 0.001')
```

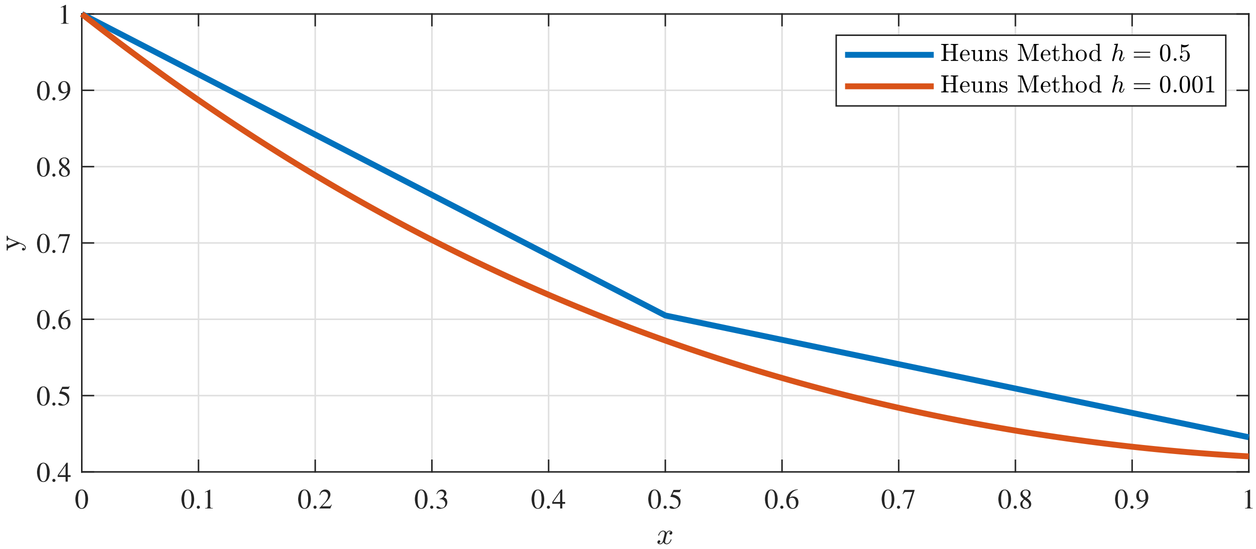


Figure 11: Plot 7

The result is plottet. The blue line is the results from the task. However to get a more accurate result, the same method was used with $h = 0.001$.

8 Task 8 Runge-Kutta 4th Order

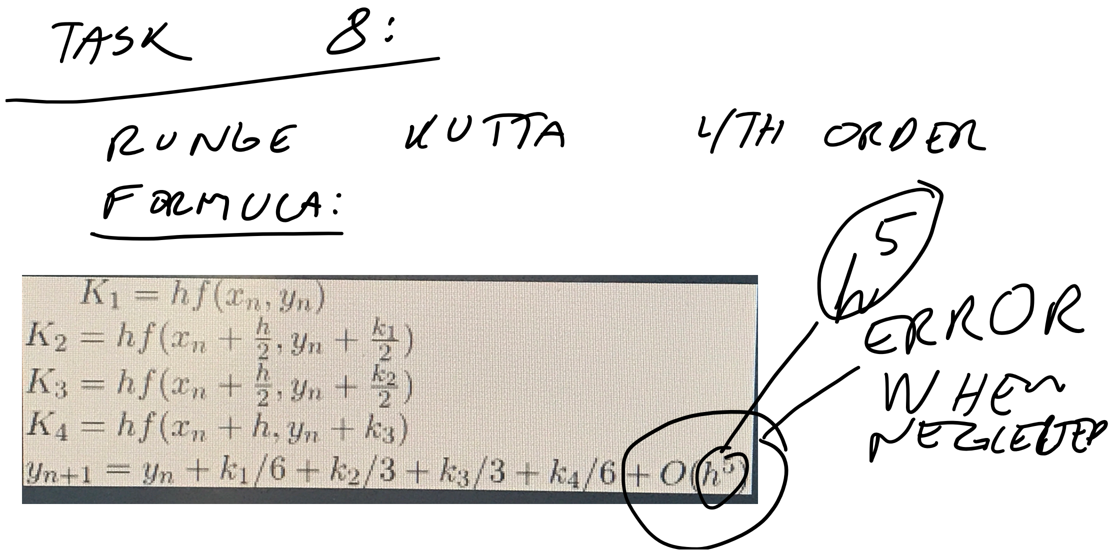


Figure 12: Task 8

```

% Task 8
clear x
clear y
 %close all; clc;
h = 0.5;
itterations = 1/h;
%initial:
i = 1;
x(i) = 0;
y(i) = 1;
% dy_dt = f(x,y) = y*x^2 - 1.2*y = y(i)*x(i)^2 - 1.2*y(i)




for i = 1 : itterations
x(i+1) = x(i) +h;
K1 = h*(y(i)*x(i)^2 - 1.2*y(i));
K2 = h*((y(i)+K1/2)*(x(i)+h/2)^2 - 1.2*(y(i)+K1/2));
K3 = h*((y(i)+K2/2)*(x(i)+h/2)^2 - 1.2*(y(i)+K2/2));
K4 = h*h*((y(i)+K3)*(x(i+1))^2 - 1.2*(y(i)+K3));
y(i+1) = y(i) + K1/6 + K2/3 + K3/3 + K4/6;
end
plot(x,y)
```

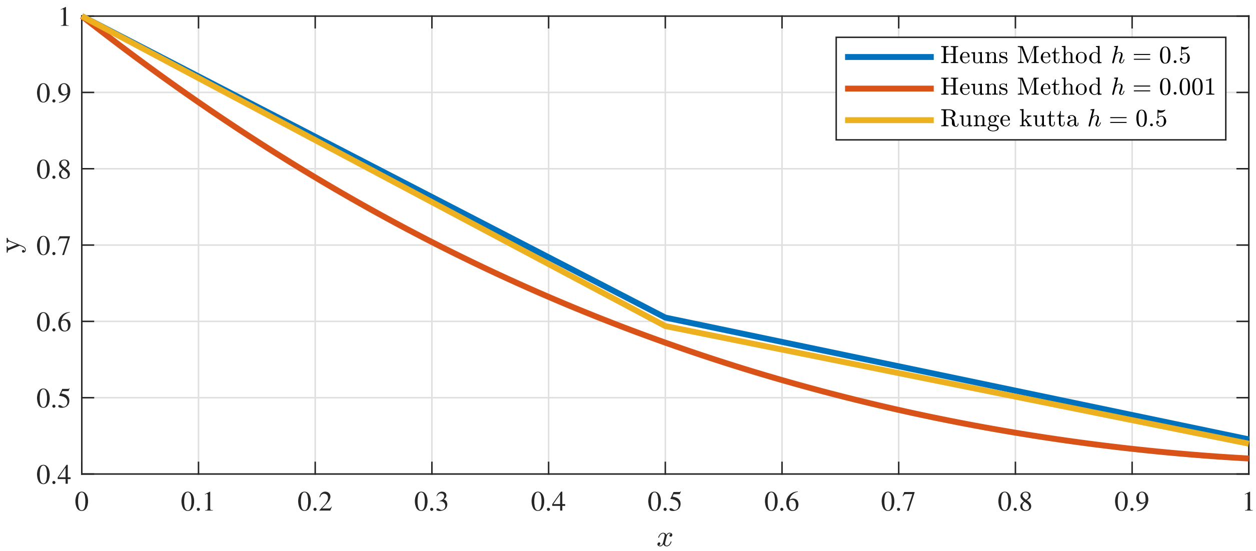


Figure 13: Plot 8

The results from the runge kutta fourth order is plotted on top of the plot from last task. Here we can see that the runge kutta is closer to the true solution.

9 Citing

Formulas from class. Heuns and runge kutta also from internett [1] and [2].

Bibliography

[1] URL: https://www.youtube.com/watch?app=desktop&v=GZMGZQhmQYM. (accessed: 08.10.2023).

[2] geeksforgeeks. *Runge-Kutta 4th Order Method to Solve Differential Equation*. URL: https://www.geeksforgeeks.org/runge-kutta-4th-order-method-solve-differential-equation/. (accessed: 08.10.2022).