**BAB I**

**PENDAHULUAN**

* 1. **Rumusan Masalah**
     1. Bagaimana cara menentukan penyelesaian Persamaan differensial Biasa secara Numerik dengan metode Euler, metode Heun, dan Metode Runge Kutta.
     2. Bagaimana cara mencari besarnya kesalahan dari suatu perhitungan solusi Persamaan Differensial Biasa secara Numerik dengan Dengan Metode Euler, Metode Heun, dan Metode Runge Kutta.
  2. **Tujuan**
     1. Dapat menentukan penyelesaian Persamaan differensial Biasa secara Numerik dengan metode Euler, metode Heun, dan Metode Runge Kutta.
     2. Mencari besarnya kesalahan dari suatu perhitungan solusi Persamaan Differensial Biasa secara Numerik dengan Dengan Metode Euler, Metode Heun, dan Metode Runge Kutta.

**BAB II**

**DASAR TEORI**

Bentuk umum suatu persamaan diferensial biasa orde n adalah :

Jika harga y dan turunan pertamanya diberikan pada t = a, maka persamaan diferensial di atas disebut problem harga awal (Initial Value Problem). Jika beberapa suku



Ditentukan pada t = a dan suku-suku lainnya pada t=b, persamaan diferensial itu disebut problem harga batas dua titik (Two-point boundary-value problem). Dalam praktikum ini diperkenalkan 3 metode popular untuk menyelesaikan persamaan diferensial yaitu metode Euler, metode Heun, dan metode Runge-Kutta orde empat dengan algoritma masing-masing dan implementasinya.

**BAB III**

**PEMBAHASAN**

* 1. **Source Code**

from math import exp

import numpy as np

# Fungsi untuk menghitung dengan menggunakan algoritma euler

def euler(ta, tb, dt, dy, y0):

# Menghitung jumlah iterasi

n = int(abs((ta - tb) / dt))

# Inisialisasi array y

y = []

y.append(y0)

# Inisialisasi array t, mulai dari ta sampai tb sebanyak n+1

t = np.linspace(ta, tb, n+1, endpoint=True)

print('x[i]\t y[i]')

for i in range(0, n):

# Menambahkan nilai y selanjutnya

y.append(y[i] + dt \* dy(t[i], y[i]))

print(round(t[i], 2), '\t : ', y[i])

print(round(t[-1], 2), '\t : ', y[-1])

return y[-1]

# Fungsi untuk menghitung dengan menggunakan algoritma heun

def heun(ta, tb, dt, dy, y0):

# Menghitung jumlah iterasi

n = int(abs((ta - tb) / dt))

# Inisialisasi array y

y = []

y.append(y0)

# Inisialisasi array t, mulai dari ta sampai tb sebanyak n+1

t = np.linspace(ta, tb, n+1, endpoint=True)

print('x[i]\t y[i]')

for i in range(0, n):

# Menghitung nilai k1 dan k2

k1 = dt \* dy(t[i], y[i])

k2 = dt \* dy(t[i], y[i] + k1)

# Menambahkan nilai y selanjutnya

y.append(y[i] + 0.5 \* (k1 + k2))

print(round(t[i], 2), '\t : ', y[i])

print(round(t[-1], 2), '\t : ', y[-1])

return y[-1]

# Fungsi untuk menghitung dengan menggunakan algoritma runge-kutta orde 4

def rungeKutta(ta, tb, dt, dy, y0):

# Menghitung jumlah iterasi

n = int(abs((ta - tb) / dt))

# Inisialisasi array y

y = []

y.append(y0)

# Inisialisasi array t, mulai dari ta sampai tb sebanyak n+1

t = np.linspace(ta, tb, n+1, endpoint=True)

print('x[i]\t y[i]')

for i in range(0, n):

# Menghitung nilai k1,k2,k3 dan k4

k1 = dt \* dy(t[i], y[i])

k2 = dt \* dy(t[i] + 0.5 \* dt, y[i] + 0.5 \* k1)

k3 = dt \* dy(t[i] + 0.5 \* dt, y[i] + 0.5 \* k2)

k4 = dt \* dy(t[i] + dt, y[i] + k3)

# Menambahkan nilai y selanjutnya

y.append(y[i] + 1/6 \* (k1 + 2 \* k2 + 2 \* k3 + k4))

print(round(t[i], 2), '\t : ', y[i])

print(round(t[-1], 2), '\t : ', y[-1])

return y[-1]

# dy/dx = x + y

def dy(x, y):

return x + y

def exact(x):

return exp(x) - x - 1

# Inisialisai Variabel

# Waktu Awal

ta = 0

# Waktu Akhir

tb = 0.10

# Nilai pada ta

y0 = 1

# Interval

dt = 0.01

# Euler

print('Menyelesaikan Persamaan Diferensial Biasa dengan Algoritma Euler\n')

y\_euler = euler(ta, tb, dt, dy, y0)

galat = abs(y\_euler - exact(tb))

print('\nNilai euler = ', y\_euler)

print('Nilai asli = ', exact(tb))

print('Dengan galat = ', galat)

# Heun

print('\nMenyelesaikan Persamaan Diferensial Biasa dengan Algoritma Heun\n')

y\_heun = heun(ta, tb, dt, dy, y0)

galat = abs(y\_heun - exact(tb))

print('\nNilai heun = ', y\_heun)

print('Nilai asli = ', exact(tb))

print('Dengan galat = ', galat)

# Heun

print('\nMenyelesaikan Persamaan Diferensial Biasa dengan Algoritma Heun\n')

y\_heun = heun(ta, tb, dt, dy, y0)

galat = abs(y\_heun - exact(tb))

print('\nNilai heun = ', y\_heun)

print('Nilai asli = ', exact(tb))

print('Dengan galat = ', galat)

# Runge-Kutta Orde 4

print('\nMenyelesaikan Persamaan Diferensial Biasa dengan Algoritma Runge-Kutta\n')

y\_rk = rungeKutta(ta, tb, dt, dy, y0)

galat = abs(y\_rk - exact(tb))

print('\nNilai runge-kutta = ', y\_rk)

print('Nilai asli = ', exact(tb))

print('Dengan galat = ', galat)

* 1. **Pembahasan Source Code**
     1. **Algoritma Euler**

Program diatas digunakan untuk menyelesaian Persamaan differensial dengan menggunakan metode euler, cara kerja program tersebut adalah Masukan Waktu awal 𝑡𝑎, Waktu akhir 𝑡𝑏, Interval 𝑑𝑡, Harga fungsi pada 𝑡𝑎, 𝑦(𝑡𝑎) = 𝑦0 Keluaran : 𝑡,𝑦.

Fungsi dy/dt dinyatakan sebagai 𝑑𝑦(𝑡,𝑦)

1. 𝑛 = (𝑡𝑎 − 𝑡𝑏)/𝑑𝑡

2. 𝑦(0) = 𝑦0, 𝑡(0) = 𝑡𝑎

3. Untuk 𝑖 = 0 sampai 𝑛 − 1

3.a. 𝑦(𝑖 + 1) = 𝑦(𝑖) + 𝑑𝑡 ∗ 𝑑𝑦(𝑡(𝑖),𝑦(𝑖))

3.b. 𝑡(𝑖 + 1) = 𝑡(𝑖) + 𝑑𝑡

4. 𝑦\_𝑒𝑢𝑙𝑒𝑟 = 𝑦(𝑛) {𝑡𝑏 = 𝑡(𝑛), y(𝑡𝑏)}

5. Selesai

**3.2.2 Algoritma Heun**

Program diatas digunakan untuk menyelesaian Persamaan differensial dengan menggunakan metode Heun, cara kerja program tersebut adalah Masukan Waktu awal 𝑡𝑎, Waktu akhir 𝑡𝑏, Interval 𝑑𝑡, Harga fungsi pada 𝑡𝑎, 𝑦(𝑡𝑎) = 𝑦0 Keluaran : 𝑡,𝑦

Fungsi dy/dt dinyatakan sebagai 𝑑𝑦(𝑡,𝑦)

1. 𝑛 = (𝑡𝑎 − 𝑡𝑏)/𝑑𝑡

2. 𝑦(0) = 𝑦0, 𝑡(0) = 𝑡𝑎

3. Untuk 𝑖 = 0 sampai 𝑛 − 1

3.a. 𝑘1 = 𝑑𝑡 ∗ 𝑑𝑦(𝑡(𝑖),𝑦(𝑖))

3.b. 𝑘2 = 𝑑𝑡 ∗ 𝑑𝑦(𝑡(𝑖),𝑦(𝑖) + 𝑘1)

3.c. 𝑦(𝑖 + 1) = 𝑦(𝑖) + 0.5 ∗ (𝑘1 + 𝑘2)

3.d. 𝑡(𝑖 + 1) = 𝑡(𝑖) + 𝑑𝑡

4. 𝑦𝑒𝑢𝑙𝑒𝑟−𝑝𝑒𝑟𝑏𝑎𝑖𝑘𝑖 = 𝑦(𝑛) {𝑡𝑏 = 𝑡(𝑛), y(𝑡𝑏)}

5. Selesai

**3.2.3 Algoritma Runge-Kutta Orde Empat**

Program diatas digunakan untuk menyelesaian Persamaan differensial dengan menggunakan metode Runge-Kutta Orde Empat, cara kerja program tersebut adalahMasukan Waktu awal 𝑡𝑎, Waktu akhir 𝑡𝑏, Interval 𝑑𝑡, Harga fungsi pada 𝑡𝑎, 𝑦(𝑡𝑎) = 𝑦0, Keluaran : 𝑡,𝑦

Fungsi dy/dt dinyatakan sebagai 𝑑𝑦(𝑡,𝑦)

1. 𝑛 = (𝑡𝑎 − 𝑡𝑏)/𝑑𝑡

2. 𝑦(0) = 𝑦0, 𝑡(0) = 𝑡𝑎

3. Untuk 𝑖 = 0 sampai 𝑛 − 1

3.a. 𝑘1 = 𝑑𝑡 ∗ 𝑑𝑦(𝑡(𝑖),𝑦(𝑖))

3.b. 𝑘2 = 𝑑𝑡 ∗ 𝑑𝑦(𝑡(𝑖) + 0.5 ∗ 𝑑𝑡,𝑦(𝑖) + 0.5 ∗ 𝑘1)

3.c. 𝑘3 = 𝑑𝑡 ∗ 𝑑𝑦(𝑡(𝑖) + 0.5 ∗ 𝑑𝑡,𝑦(𝑖) + 0.5 ∗ 𝑘2)

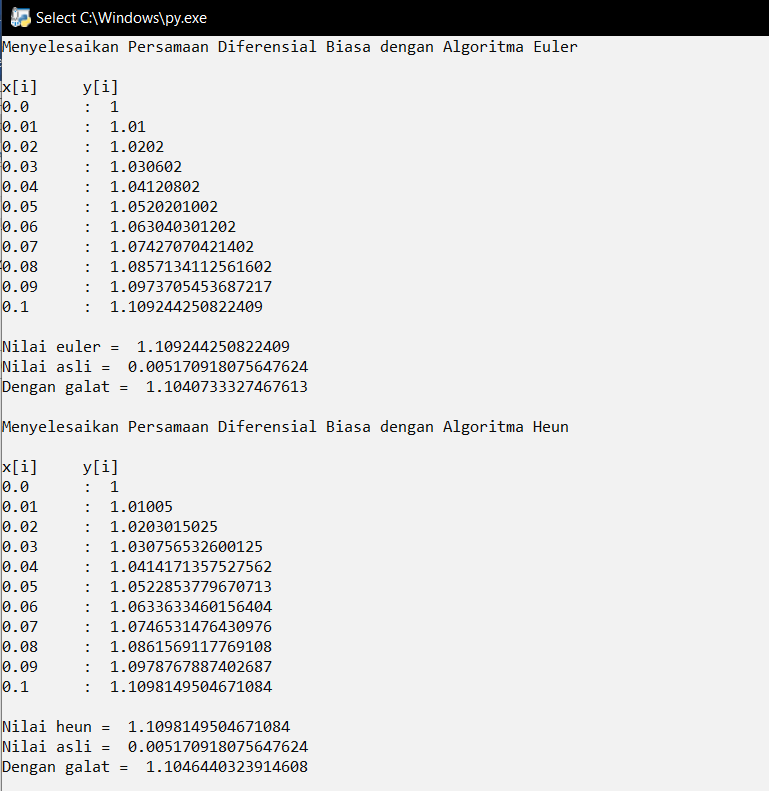
3.d. 𝑘4 = 𝑑𝑡 ∗ 𝑑𝑦(𝑡(𝑖) + 𝑑𝑡,𝑦(𝑖) + 𝑘3)

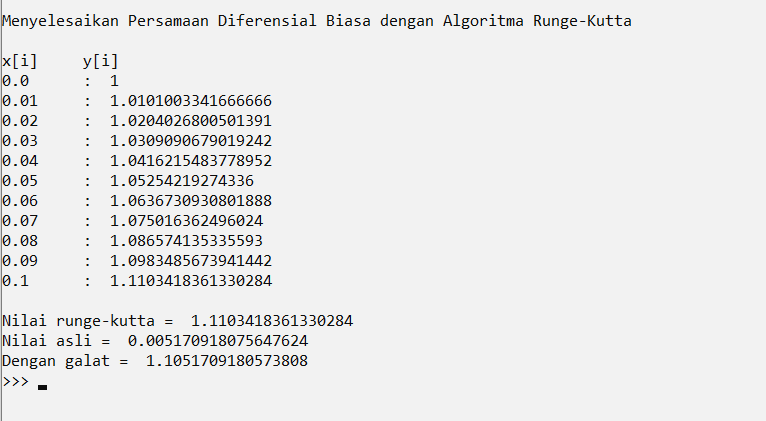
3.e. 𝑦(𝑖 + 1) = 𝑦(𝑖) + 1/6 ∗ (𝑘1 + 2 ∗ 𝑘2 + 2 ∗ 𝑘3 + 𝑘4)

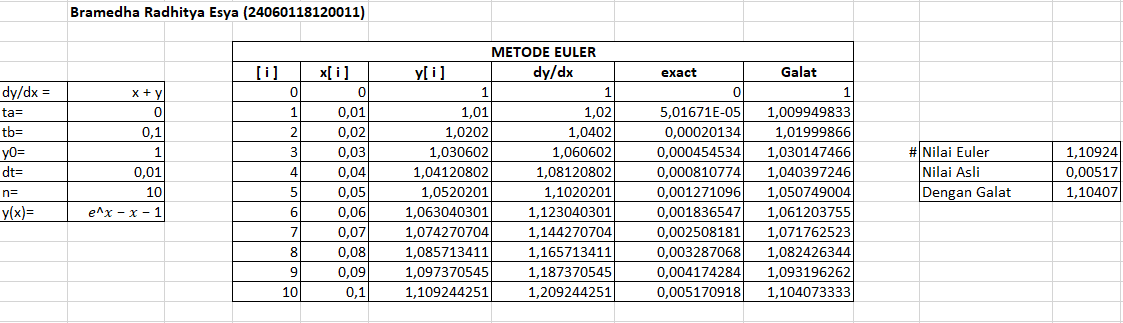
3.f. 𝑡(𝑖 + 1) = 𝑡(𝑖) + 𝑑𝑡

4. 𝑦\_𝑟𝑢𝑛𝑔𝑒𝑘𝑢𝑡𝑡𝑎\_𝑜𝑟𝑑𝑒4 = 𝑦(𝑛) {𝑡𝑏 = 𝑡(𝑛), y(𝑡𝑏)}

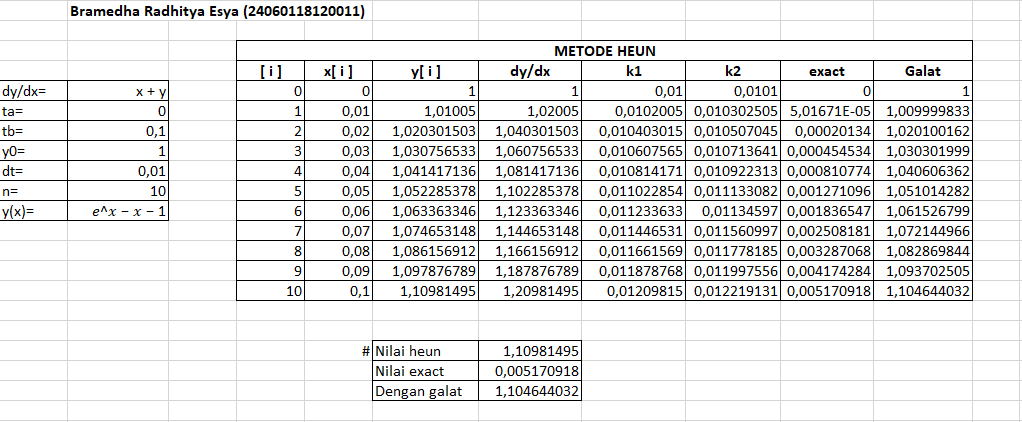
5. Selesai

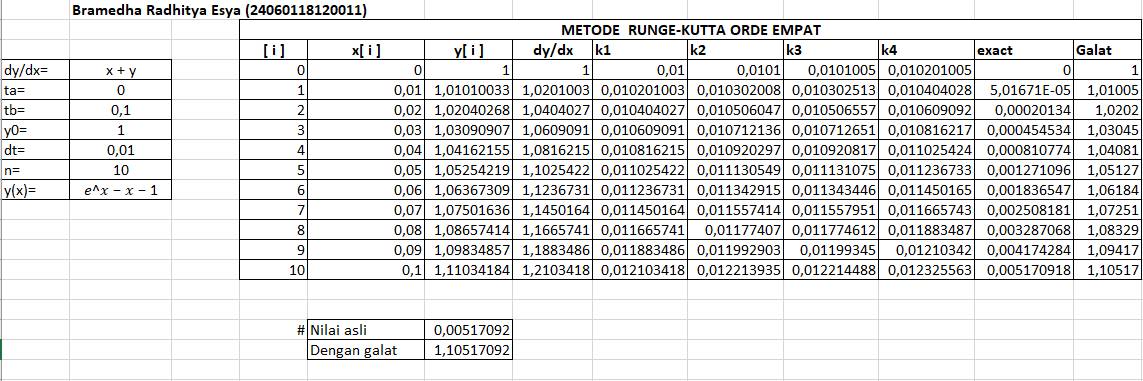
* 1. **Screenshot program**



* 1. **Screenshot excel**

Screenshot Metode Euler

Screenshot Metode Heun

 Screenshot Metode Runge-Kutta Orde Empat

**3.5 Pembahasan Excel**

Bentuk umum suatu persamaan diferensial biasa orde n adalah :

Jika harga y dan turunan pertamanya diberikan pada t = a, maka persamaan diferensial di atas disebut problem harga awal (Initial Value Problem). Jika beberapa suku



Ditentukan pada t = a dan suku-suku lainnya pada t=b, persamaan diferensial itu disebut problem harga batas dua titik (Two-point boundary-value problem). Dalam praktikum ini diperkenalkan 3 metode popular untuk menyelesaikan persamaan diferensial yaitu metode Euler, metode Heun, dan metode Runge-Kutta orde empat dengan algoritma masing-masing dan implementasinya.

**BAB IV**

**PENUTUP**

**4.1. Kesimpulan**

Dengan menggunakan metode Euler, Heun, Ruge-Kutta orde 4. Kita dapat menyelesaikan berbagai kondisi pada persamaan differensial biasa secara numerik. dengan dilakukannya perhitungan secara langsung oleh komputer, dapat disimpulkan bahwa, :

1. nilai dari algoritma euler adalah : 1.109244250822409  
   dengan nilai asli : 0.005170918075647624   
   galat : 1.1040733327467613
2. Nilai dari algoritma heun adalah : 1.1098149503471084  
   Dengan nilai asli : 0.005170918075647624  
   Galat : 1.1046440323914608
3. Nilai dari algoritma runge-kutta orde 4 adalah : 1.1103418361330284  
   Dengan nilai asli : 0.005170918075647624  
   Galat : 1.1051709080573808

**DAFTAR PUSTAKA**

Munir, Rinaldi. 2015. *Metode Numerik.* Bandung : Informatika

Sasongko, Priyo Sidik dan Suhartono. 2019. Modul Praktikum Metode Numerik. Semarang: Departemen Informatika/Ilmu Komputer Fakultas Sains dan Matematika Universitas Diponegoro