Lukito Andriansyah

An Applied of Euler-Methods in Basic Physics: Ball Thrown Falling Down

Personal Project for Data Analysis

Basic Physics I

lukitoandriansyah45@gmail.com

Ringkasan

Tentu familiar dengan kasus bola yang dilemparkan jatuh ke bawah. Biasanya, variabel yang dikenal seperti waktu (t), kecepatan (v), jarak (x), dan percepatan (a), digunakan untuk menyelesaikan kasus ini dalam kondisi ideal tanpa menganggap adanya *drag force*. Jika keberadaan *drag force* dipertimbangkan, akan ada kondisi yang membuat bola mengalami percepatan nol, Ini disebut sebagai laju terminal (v_T). Salah satu cara untuk mencarinya adalah dengan menggunakan *Euler-Methods*. Proyek ini bertujuan menerapkan *Euler-Methods* untuk memperkirakan laju bola yang dilemparkan ke bawah dan waktu yang diperlukan untuk mencapai 99% dari laju terminal. Data diperoleh dari tugas *Numerical Integration* dalam buku Fisika Dasar. Kemudian, data disimpan dan diproses di Pycham Community Edition 2021.3.2 . Hasilnya menunjukkan bahwa untuk kasus bola dijatuhkan dengan laju awal 35 km/jam, setelah 10 detik bola akan memiliki laju 41.2035 m/s. Pada kasus lain, yaitu bola dilepas tanpa laju awal, menunjukkan bola membutuhkan sekitar 11.23923 s setelah dilepaskan untuk mencapai 99% dari kecepatan terminal.

Kata Kunci*: Drag Force, Euler-Methods,* Laju Terminal*, Numerical Integration*

Lukito Andriansyah | **lukitoandriansyah45@gmail.com**

Pendahuluan

Dalam fisika, tentu kita sudah mengetahui bahwa, jika sebuah partikel bergerak di bawah pengaruh gaya konstan, tentu percepatannya akan konstan dan kita dapat menemukan kecepatan beserta posisinya dari rumus kinematika pada percepatan konstan. Sekarang, mari kita coba bayangkan sebuah partikel yang sedang bergerak melalui suatu tempat dimana terdapat gaya, sehingga kini percepatannya bergantung juga pada posisi dan kecepatan benda itu sendiri. Pada keadan ini, kita dapat katakan bahwa posisi, kecepatan, dan percepatan partikel pada suatu waktu tertentu akan menentukan posisi, kecepatan, dan percepatan pada saat berikutnya.

Untuk menyelesaikan kasus tersebut, kita perlu mengerti bahwa posisi, kecepatan, dan percepatan suatu benda akan terus menerus berubah terhadap waktu. Sehingga kita dapat memperkirakan dengan mengganti variasi selang waktu dengan nilai yang kecil. Ide paling sederhananya adalah mengasumsikan bahwa percepatan akan konstan selama selang waktu tersebut. Pendekatan ini dikenal sebagai *Euler-Methods*. Jika selang waktu cukup kecil, perubahan percepatan selama selang waktu tersebut akan kecil dan dapat diabaikan (Tipler & Mosca, 2007).

Proyek ini dilakukan untuk memperkirakan laju bola yang dilempar ke bawah dan waktu yang dibutuhkan untuk mencapai laju terminal. Data diperoleh dari soal *Numerical Integration* buku fisika dasar yang kemudian disimpan dan diolah dalam menggunakan Pycham Community Edition 2021.3.2.

Teori Singkat

Numerical Integration

Misalkan diketahui posisi x_0 , kecepatan v_0 , dan percepatan a_0 partikel pada suatu waktu awal t_0 . Jika kita mengabaikan perubahan kecepatan selama selang waktu Δt , posisi baru x_1 diberikan oleh

$$x_1 = x_0 + v_{0x}\,\Delta t \qquad (1)$$

Demikian pula, jika kita asumsikan percepatan konstan selama Δt , kecepatan pada waktu $t_1 = t_0 + \Delta t$ diberikan oleh

$$v_{1x} = v_{0x} + a_{0x}\,\Delta t \qquad (2)$$

Kita dapat menggunakan nilai x_1 dan v_{1x} untuk menghitung percepatan baru a_{1x} menggunakan hukum kedua Newton, dan kemudian gunakan x_1 , v_{1x} , dan a_{1x} untuk menghitung x_2 dan v_{2x} .

$$x_2 = x_1 + v_{1x}\,\Delta t \qquad (3)$$
$$v_{2x} = v_{1x} + a_{1x}\,\Delta t \qquad (4)$$

Lukito Andriansyah | lukitoandriansyah45@gmail.com

Hubungan antara posisi dan kecepatan pada waktu t_n dan waktu $t_{n+1} = t_n + \Delta t$ diberikan

$$x_{n+1} = x_n + v_{nx}\,\Delta t \quad (5)$$
$$v_{(n+1)x} = v_{nx} + a_{nx}\,\Delta t \quad (6)$$

Untuk mencari kecepatan dan posisi pada suatu waktu t , maka kita membagi selang waktu $t - t_0$ menjadi banyak interval yang lebih kecil dari t dan menerapkan Persamaan (5) dan (6), dimulai pada waktu awal t_0 . Hal ini akan melibatkan sejumlah besar perhitungan sederhana dan berulang yang paling mudah dilakukan di komputer. Teknik memecah interval waktu menjadi langkah-langkah kecil dan menghitung percepatan, kecepatan, dan posisi pada setiap langkah menggunakan nilai dari langkah sebelumnya disebut *numerical integration* (Tipler & Mosca, 2007).

Benda Jatuh

Tinjaulah sebuah benda yang sedang terjatuh dari suatu ketinggian. Benda tersebut berada dalam pengaruh gaya gravitasi dan juga *drag force* selama tejatuh yang mana nilai *drag force* proporsional dengan laju kuadrat (Cutnell JD, 2018). Dari kasus tersebut, kita ingin menemukan kecepatan dan jarak tempuh sebagai fungsi waktu. Persamaan gerak benda yang terjatuh adalah sebagai berikut:

$$mg - bv^2 = ma_x$$

dimana arah kebawah adalah positif. Sehingga percepatan diberikan sebagai:

$$a_x = g - \frac{b}{m}v^2 \quad (7)$$

Ambil nilai untuk $a_x = 0$, maka persamaan tersebut menjadi:

$$0 = g - \frac{b}{m}v^2$$
$$\frac{b}{m} = \frac{g}{v^2}$$

Variabel v^2 pada keadaan ini (percepatan nol) dituliskan sebagai laju terminal kuadrat ($v_T{}^2$). Jika nilai tersebut disulih ke dalam persamaan (7) akan memberikan:

$$a_x = g\left(1 - \frac{v^2}{v_T{}^2}\right) \quad (8)$$

Lukito Andriansyah | lukitoandriansyah45@gmail.com

Metode

Alat

Proyek ini dilakukan secara mandiri dengan memanfaatkan Laptop, dan Pycham Community Edition 2021.3.2.

Data

Data proyek diambil dari soal latihan yang berisikan informasi variabel diketahui sebagai berikut:

- Kasus 1
  1. Bola dilempar ke bawah demgam laju awal $v_0 = 35$ km/jam
  2. Laju terminal bola adalah $v_T = 150$ km/jam
  3. Model *Drag Force* yang digunakan adalah $f = bv^2$
  4. Laju estimasi dihitung setelah $t = 10$ s
- Kasus 2
  1. Bola dilepas dari ketinggian tertentu tanpa laju awal
  2. Laju terminal bola adalah $v_T = 150$ km/jam
  3. Model *Drag Force* yang digunakan adalah $f = bv^2$
  4. Waktu yang dihitung adalah setelah bola mencapai 0.99 v_T

Persiapan Data

Oleh karena kita hanya memiliki data yang sangat terbatas, maka kita dapat mengembangkan data tersebut melalui persamaan (5), (6), dan (8). Tetapi sebelum itu, kita perlu melakukan beberapa hal agar data yang kita miliki nanti bis di analisis. Berikut adalah yang perlu dilakukan:

A. Pembuatan DataBase

Pembuatan DataBase dilakukan untuk menyimpan hasil pengolahan ataupun data agar lebih mudah untuk dilakukan analisis. Pada kasus ini, saya membuat DataBase di MySQL dengan nama DataBase 'Euler_Methods'. Adapun kode berikut dapat digunakan untuk membuatnya:

```
import mysql.connector as msql
from mysql.connector import Error
try:
    conn = msql.connect(host='***', user='***',
                        password='***')
    if conn.is_connected():
        cursor = conn.cursor()
        cursor.execute("CREATE DATABASE Euler_Methods")
        print("Euler_Methods database is created")
except Error as e:
    print("Error while connecting to MySQL", e)
```

catatan: untuk bagian yang berisi ***, silakan disesuaikan dengan karakter MySQL masing-masing.

Lukito Andriansyah | lukitoandriansyah45@gmail.com

B. Pembuatan DataFrame

Oleh karena Data yang akan kita analisis dalam kasus ini memiliki lebih dari satu kolom, tentu akan mudah jika kita bentuk data menjadi dalam bentuk per kolom. Kumpulan data dalam tiap kolom ini dikenal sebagai DataFrame (Hamami, 2020). Terdapat Tiga buah DataFrame yang akan dibuat, masing- masing adalah DataFrame Kasus 1, DataFrame Kasus 1 dengan variasi selang waktu berbeda, dan DataFrame Kasus 2. Berikut adalah *Code* yang dapat digunakan untuk membuat ketiga DataFrame tersebut:

***Code* 1 DataFrame kasus 1**

```
import pandas as pd
import numpy as np

"""Known Variables in SI units for Case 1"""
v_0 = 35 * (1000/3600)
v_term = 150 * (1000/3600)
t_total = float(10)
t_df = t_total + 0.01

"""
Using Methode Euler
"""
t_0 = 0
x_0 = 0
a_0 = float(9.81)
n = 5000
dt = 6 * t_df / n

#Step1: Membuat kerangka data frame
data =[t_0, x_0, v_0]
data.append(a_0*(1-(data[2]**2/v_term**2)))
rows = [0]
while n not in rows:
    rows.append(rows[-1] + 1)
index = np.array(rows)
df = pd.DataFrame(columns=['t','x','v','a'], index=
index)
df.loc[0] = data

#Step2: Melengkapi Kerangka Dataframe
i = 1
while np.isnan(df['t'].loc[i]).sum() > 0 and i < n:
    df['v'].loc[i] = df['v'].loc[i - 1] +
(df['a'].loc[i - 1] * dt)
    df['x'].loc[i] = df['x'].loc[i - 1] +
(df['v'].loc[i - 1] * dt)
    df['t'].loc[i] = df['t'].loc[i-1]+dt
    df['a'].loc[i] = a_0 * (1 - (df['v'].loc[i] ** 2 /
v_term ** 2))
    i += 1
df_euler_methods = df.dropna(axis=0)
df_euler_methods.to_csv('Data Frame Nomor 1_a_1.csv',
index=False, encoding='utf-8', quoting=1)
```

***Code* 2 DataFrame kasus 1 selang waktu berbeda**

Lukito Andriansyah | lukitoandriansyah45@gmail.com

```
import pandas as pd
import numpy as np
from Data_1_a_1 import n

"""
Known Variables in SI units for Case 1
with different time interval
"""
v_0 = 35 * (1000/3600)
v_term = 150 * (1000/3600)
t_total = float(10)
t_df = t_total + 0.01


"""
Using Methode Euler
"""
t_0 = 0
x_0 = 0
a_0 = float(9.81)
dt = (6 * t_df) / (2 * n)
l = int(6 * t_df / dt)

#Step1: Membuat kerangka data frame
data =[t_0, x_0, v_0]
data.append(a_0*(1-(data[2]**2/v_term**2)))
rows = [0]
while l not in rows:
    rows.append(rows[-1] + 1)
index = np.array(rows)
df = pd.DataFrame(columns=['t','x','v','a'], index=
index)
df.loc[0] = data

#Step2: Melengkapi Kerangka Dataframe
i = 1
while np.isnan(df['t'].loc[i]).sum() > 0 and i < l:
    df['v'].loc[i] = df['v'].loc[i - 1] +
(df['a'].loc[i - 1] * dt)
    df['x'].loc[i] = df['x'].loc[i - 1] +
(df['v'].loc[i - 1] * dt)
    df['t'].loc[i] = df['t'].loc[i-1]+dt
    df['a'].loc[i] = a_0 * (1 - (df['v'].loc[i] ** 2 /
v_term ** 2))
    i += 1
df_euler_methods = df.dropna(axis=0)
df_euler_methods.to_csv('Data Frame Nomor 1_a_2.csv',
index=False, encoding='utf-8', quoting=1)
```

Catatan: Data_1_a_1 pada Code di atas merupakan nama file .py yang berisikan Code 1

Lukito Andriansyah | lukitoandriansyah45@gmail.com

***Code* 3 DataFrame kasus 2**

```
import math as m
import pandas as pd
import numpy as np
from Data_1_a_1 import df_euler_methods
from Data_1_a_1 import dt
from Data_1_a_1 import n

"""
Known Variables in SI units
Ball released at rest (Case 2)
"""
v_0 = 0
v_term = 150 * (1000/3600)

"""
Using Methode Euler
"""
t_0 = 0
x_0 = 0
a_0 = float(9.81)
t_total = m.sqrt(2*df_euler_methods['x'].loc[n-1]/a_0)
t_df = t_total + 0.01
dt = dt
z = 2*n

#Step1: Membuat kerangka data frame
data =[t_0, x_0, v_0]
data.append(a_0*(1-(data[2]**2/v_term**2)))
rows = [0]
while z not in rows:
    rows.append(rows[-1] + 1)
index = np.array(rows)
df = pd.DataFrame(columns=['t','x','v', 'a'], index=
index)
df.loc[0] = data

#Step2: Melengkapi Kerangka Dataframe
i = 1
while np.isnan(df['t'].loc[i]).sum() > 0 and i < z:
    df['v'].loc[i] = df['v'].loc[i - 1] +
(df['a'].loc[i - 1] * dt)
    df['x'].loc[i] = df['x'].loc[i - 1] +
(df['v'].loc[i - 1] * dt)
    df['t'].loc[i] = df['t'].loc[i-1]+dt
    df['a'].loc[i] = a_0 * (1 - (df['v'].loc[i] ** 2 /
v_term ** 2))
    i += 1
df_euler_methods = df.dropna(axis=0)
df_euler_methods.to_csv('Data Frame Nomor 1_b_1.csv',
index=False, encoding='utf-8', quoting=1)
```

Catatan: Data_1_a_1 pada Code di atas merupakan nama file .py yang berisikan Code 1

Lukito Andriansyah | lukitoandriansyah45@gmail.com

Penjelasan *Code*:

Bila diperhatikan dari ketiga *Code* tersebut, untuk membuat DataFrame sendiri terdiri dari dua tahap, yaitu pembuatan kerangka DataFrame dan kemudian melengkapi isi DataFrame. Perlu diingat, dua step ini tidak baku, dengan kata lain, dapat disesuaikan dengan kebutuhan masing-masing. Karena umumnya ada banyak cara yang dapat digunakan untuk membuat sebuah DataFrame (SkillPlus, 2021).

Pada tahap pembuatan kerangka DataFrame, ide utama yang ingin disampaikan adalah bagaimana cara kita menyediakan ruang kosong yang jumlahnya secara otomatis menyesuaiakan dengan apa yang diminta.

Mari kita cuplik berdasarkan kasus 1. Oleh karena hanya diketahui variabel laju awal dan laju terminal, maka kita dapat menuliskan bahwa waktu awal dan jarak tempuh awal sebagai 0. Selain itu, kita juga dapat menuliskan nilai gravitasi yang dideklarasikan sebagai a_0. Nilai-nilai tersebut akan menjadi isi dari baris petama dalam kerangka DataFrame dan selanjutnya akan diikuti dengan pembuatan baris baru DataFrame tanpa adanya nilai (NaN) (FooBar). Oleh karena itu, dibuatlah *Code* seperti cuplikan berikut dalam *Code* 1:

```
#Step1: Membuat kerangka data frame
data =[t_0, x_0, v_0]
data.append(a_0*(1-(data[2]**2/v_term**2)))
rows = [0]
while n not in rows:
    rows.append(rows[-1] + 1)
index = np.array(rows)
df = pd.DataFrame(columns=['t','x','v','a'], index=
index)
df.loc[0] = data
```

Selanjutnya adalah melengkapi DataFrame. Berdasarkan teori yang telah dikemukakan, maka kita dapat melengkapi masing-masing NaN untuk kolom t, x, v, dan a melalui hubungan persamaan (5), (6), dan (8). Oleh karena itu, dibuatlah *Code* seperti cuplikan berikut dalam *Code* 1:

```
i = 1
while np.isnan(df['t'].loc[i]).sum() > 0 and i < z:
    df['v'].loc[i] = df['v'].loc[i - 1] +
(df['a'].loc[i - 1] * dt)
    df['x'].loc[i] = df['x'].loc[i - 1] +
(df['v'].loc[i - 1] * dt)
    df['t'].loc[i] = df['t'].loc[i-1]+dt
    df['a'].loc[i] = a_0 * (1 - (df['v'].loc[i] ** 2 /
v_term ** 2))
    i += 1
df_euler_methods = df.dropna(axis=0)
df_euler_methods.to_csv('Data Frame Nomor 1_b_1.csv',
index=False, encoding='utf-8', quoting=1)
```

Dua baris *Code* terakhir akan menyimpan hasil DataFrame dalam file berformat CSV.

C. *Load* DataFrame ke Database

Setelah membuat DataFrame, tentuk untuk mempermudah analisis kita perlu melakukan *load* data ke DataBase. Untuk ketiga DataFrame tersebut, cara yang digunakan sama. Hanya terdapat beberapa modifikasi dalam penamaan tabel DataFrame. Salah satu cuplikan *Code* yang digunakan adalah sebagai berikut:

***Code* 4 Contoh load ke DataBase**

```
# import the module
from sqlalchemy import create_engine
import pymysql
import mysql
# create sqlalchemy engine
engine =
create_engine("mysql+pymysql://{user}:{pw}@localhost/{
db}"
                      .format(user="***", pw="***",
                      db="Euler_Methods"))
# Insert whole DataFrame into MySQL
df_euler_methods.to_sql('###', con = engine, if_exists
= 'append', chunksize = 1000,index=False)
```

catatan: untuk bagian yang berisi ***, silakan disesuaikan dengan karakter MySQL masing-masing. Sementara bagian yang berisi ###, adalah nama tabel yang* dapat *disesuaikan berdasarkan kebutuhan

Nama tabel yang digunakan dalam kasus ini untuk ketiga DataFrame masing-masing adalah: **Euler_Method_No_1_a_1 , Euler_Method_No_1_a_2 , dan Euler_Method_No_1_b_1.**

Melakukan import DataFrame ke dalam DataBase menggunakan .to_sql() jauh lebih efektif dan efisien dibandingkan cara manual (Ichi.Pro)**.**

Pengolahan Data

Terdapat dua kasus yang akan kita selesaikan, yaitu kasus 1 dan kasus 2.

1. Kasus 1

Secara umum, kita hanya perlu menentukan estimasi laju benda setelah 10 s. Akan tetapi, mungkin akan lebih baik jika kita juga meninjau besarnya ketidakpastian/*Error* estimasi dan juga keadaan gerak benda (t, x, v, dan a) saat mencapai laju terminal. Sehingga kita kini memiliki **tiga sub-pertanyaan** yaitu **estimasi laju setelah 10 s, % *Error* estimasi laju tersebut dan keadaan gerak saat mencapai laju terminal.**

Lukito Andriansyah | lukitoandriansyah45@gmail.com

a. Estimasi Laju Setelah 10 s

Untuk melihat estimasi laju, kita dapat gunakan *Code* berikut:

***Code* 5 Estimasi laju setelah 10 s**

```
import mysql.connector as msql
# Execute query
conn = msql.connect(host='***', user='***',
                            password='***',
db="Euler_Methods")
cursor = conn.cursor()
v_after_10_s = "SELECT distinct " \
               "em1a1.v " \
               "FROM " \
               "euler_methods.euler_method_no_1_a_1 as
em1a1, " \
               "euler_methods.euler_method_no_1_a_2 as
em1a2 " \
               "WHERE ( em1a1.t >= 10 and em1a2.t >= 10
)" \
               "and round(em1a1.t,5) = round(em1a2.t,5)
" \
               "limit 1"
cursor.execute(v_after_10_s)

# Fetch all the records
result = cursor.fetchall()
for i in result:
    print(i)
```

catatan: untuk bagian yang berisi ***, silakan disesuaikan dengan karakter MySQL masing-masing.

Jika *Code* tersebut dijalankan, akan menampilkan hasil:

```
"D:\Python\Python Inst
(41.20349480771995,)
```

Gambar 1 *Output* estimasi laju setelah 10 s

b. %*Error* Laju Estimasi Setelah 10 s

Untuk menghitung %*Error* dari laju estimasi tersebut, kita dapat membandingkannya dengan DataFrame yang yang selang waktunya dibuat lebih kecil dari nilai tersebut (Tipler & Mosca, 2007). Untuk itu, *Code* berikut dapat digunakan untuk mengetahui %*Error* yang dimaksud:

***Code* 6 %*Error* laju estimasi**

```
"""
Berapa Ketidakpastian kecepatan setalah 10 detik
berdasarkan dua data tersebut?
"""
```


Lukito Andriansyah | lukitoandriansyah45@gmail.com

```
import mysql.connector as msql
# Execute query
conn = msql.connect(host='***', user='***',
                    password='***',
db="Euler_Methods")
cursor = conn.cursor()
v_after_10_s = "SELECT distinct " \
               "em1a1.v, em1a2.v, abs(1-
(em1a2.v/em1a1.v))*100 as KetidakPastian " \
               "FROM " \
               "euler_methods.euler_method_no_1_a_1 as
em1a1, " \
               "euler_methods.euler_method_no_1_a_2 as
em1a2 " \
               "WHERE (em1a1.t >= 10 and em1a2.t >= 10)
" \
               "AND round(em1a1.t,5)=round(em1a2.t,5) "
\
               "limit 1"
cursor.execute(v_after_10_s)
# Fetch all the records
result = cursor.fetchall()
for i in result:
    print(i)
```

catatan: untuk bagian yang berisi ***, silakan disesuaikan dengan karakter MySQL masing-masing.

Jalankan program tersebut dan akan muncul hasil berikut ini:

```
"D:\Python\Python Installer\python.exe" "D:/Python/pycharm inst
(41.20349480771995, 41.20308677340373, 0.0009902905521053107)

Process finished with exit code 0
```

Gambar 2 *Output* %*Error* laju estimasi

Urutan hasil *Code* di atas dari kiri ke kanan adalah nilai estimasi laju kasus 1, nilai estimasi laju kasus satu dengan interval waktu lebih kecil, dan % *Error* estimasi laju.

c. Keadaan Gerak Saat Mencapai Laju Terminal

Untuk melihat keadaan gerak ini, kita dapat menggunakan *Code* berikut:

***Code* 7 Keadaan gerak saat laju terminal**

```
"""
Keadaan gerak setelah mencapai laju terminal?
"""
import mysql.connector as msql
# Execute query
conn = msql.connect(host='***', user='***',
```

```
                            password='***',
db="Euler_Methods")
cursor = conn.cursor()
coord_v_term = "SELECT t, x, v, a " \
               "FROM Euler_Method_No_1_a_1 " \
               "WHERE v >= 150*(1000/3600) " \
               "Limit 1"
cursor.execute(coord_v_term)
# Fetch all the records
result = cursor.fetchall()
for i in result:
    print(i)
```

catatan: untuk bagian yang berisi *, silakan disesuaikan dengan karakter MySQL masing-masing.**

Hasil dari *Code* tersebut jika dijalankan adalah:

```
"D:\Python\Python Installer\python.exe" "D:/Python/pycharm installer/Project 5/Euler Me
('42.19215000000272', '1672.4106802511492', 41.66666655008769, 5.489471131658874e-08)

Process finished with exit code 0
```

Gambar 3 *Output* keadaan gerak saat laju terminal

2. Kasus 2

Pada kasus 2 ini, kita hanya perlu melihat **keadaaan gerak benda ketika mencapai laju sebesar 0.99 dari laju terminal**. Berikut *Code*-nya:

***Code* 8 Keadaan gerak saat 0.99 laju terminal**

```
"""
Keadaan gerak saat 0.99 laju terminal?
"""
import mysql.connector as msql
# Execute query
conn = msql.connect(host='***', user='***',
                    password='***', db="Euler_Methods")
cursor = conn.cursor()
t_when_v_is_v_term = "SELECT t, x, v, a " \
                     "FROM Euler_Method_No_1_b_1 " \
                     "WHERE v >= 0.99 * 150 * (1000/3600) " \
                     "limit 1"
cursor.execute(t_when_v_is_v_term)
# Fetch all the records
result = cursor.fetchall()
for i in result:
    print(i)
```

catatan: untuk bagian yang berisi *, silakan disesuaikan dengan karakter MySQL masing-masing.**

Hasil yang akan ditampilkan adalah sebagai berikut:

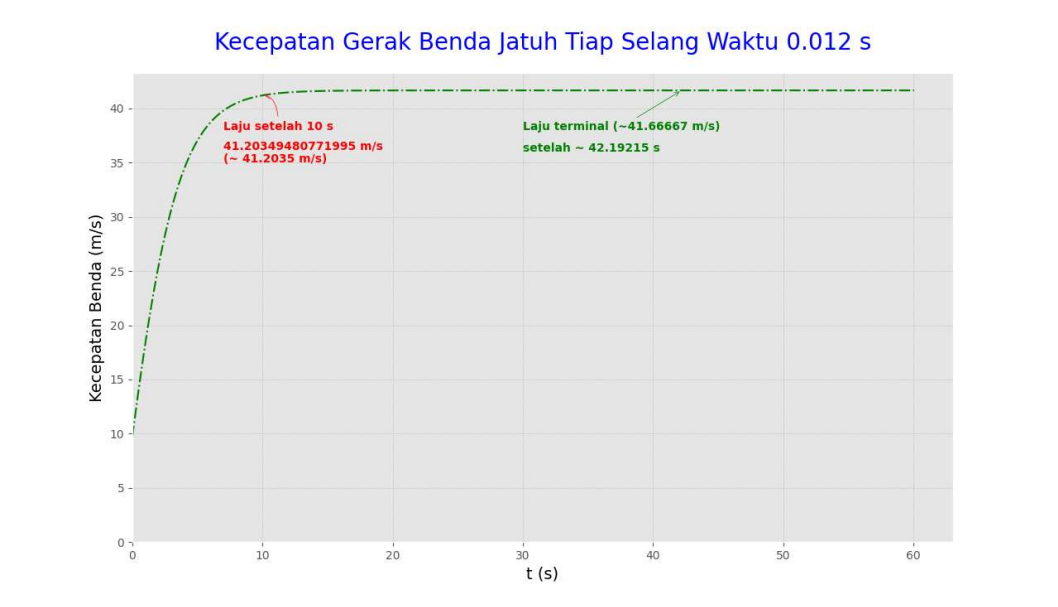
```
"D:\Python\Python Installer\python.exe" "D:/Python/pycharm installer/Project 5/Euler Me
('11.239227999998855', '346.5026117569176', '41.250360421245865', 0.19505098126128512)

Process finished with exit code 0
```

Gambar 4 *Output* keadaan gerak saat 0.99 laju terminal

Lukito Andriansyah | lukitoandriansyah45@gmail.com

Hasil

Untuk kasus 1 (selang waktu 0,012 s), diperoleh hasil bahwa setelah bola yang dilempar ke bawah dengan laju 35 km/jam, 10 s kemudian bola akan mencapai laju 41,2035 m/s. *Error* dalam angka tersebut sangat kecil, yakni 0,00099 %.



Gambar 5 Grafik v vs. t untuk kasus 1 (selang waktu 0,012 s)

Ketika mencapai laju terminal 150 km/jam (41,66667 m/s), bola memiliki nilai percepatan sangat kecil (~ 0 m/s^2) berdasarkan batas maksimum ordo 10^{-5} (Tipler & Mosca, 2007). Sehingga laju bola akan konstan. Keadaan ini tercapai setelah bola bergerak selama 14,19215 s dan menempuh jarak sejauh 1672,41068 m atau setara 1,672 km.

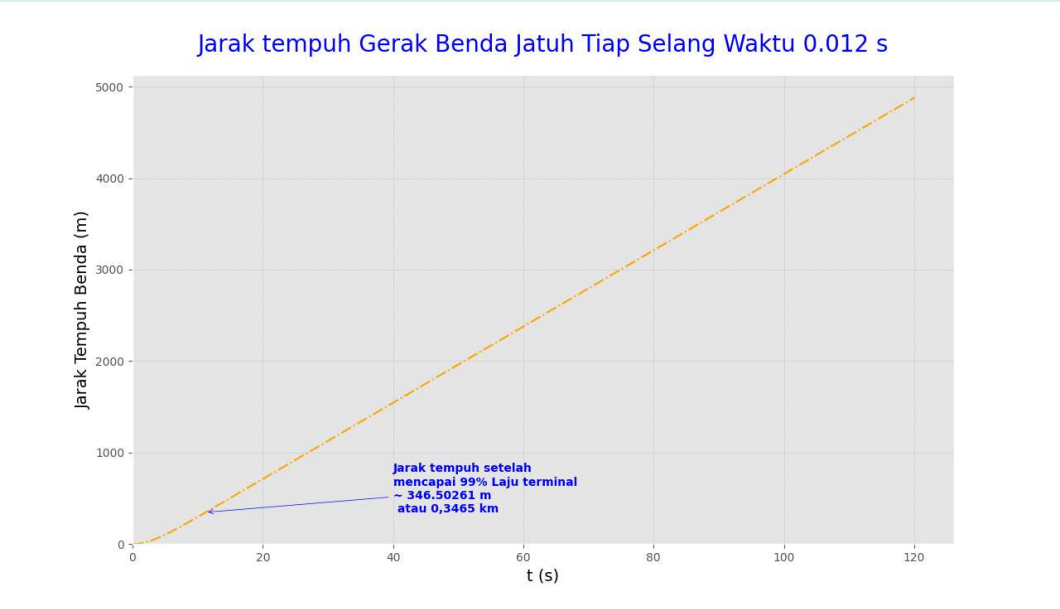
Gambar 6 Grafik x vs. t untuk kasus 1 (selang waktu 0,012 s)

Lukito Andriansyah | lukitoandriansyah45@gmail.com

Untuk kasus 2 (selang waktu 0,012 s), setelah bola dilepaskan tanpa laju awal, bola akan mencapai 99% laju terminal (41,25036 m/s) setelah bergerak selama 11,23923 s.

Gambar 7 Grafik a vs. t dan v vs. t untuk kasus 2 (selang waktu 0,012 s)

Pada keadaan ini, bola telah menempuh jarak sejauh 346,50261 m atau 0,3465 km dengan percepatan yang dialaminya sebesar 0,19505 dalam SI.



Gambar 8 Grafik x vs. t untuk kasus 2 (selang waktu 0,012 s)

Lukito Andriansyah | lukitoandriansyah45@gmail.com

Referensi

Cutnell JD, J. K. (2018). *Physics Eleventh Edition.* New York: Wiley.

FooBar. (t.thn.). */insert-a-row-to-pandas-dataframe*. Dipetik Mei 21, 2022, dari qastack.id: https://qastack.id/programming/24284342/insert-a-row-to-pandas-dataframe

Hamami, F. (2020, Juni 01). *Python Pandas: Pandas DataFrame*. Dipetik Mei 22, 2022, dari ngodingdata.com: https://ngodingdata.com/python-pandas-dataframe/

Ichi.Pro. (t.thn.). *cara-mengimpor-file-csv-ke-database-mysql-menggunakan-python*. Dipetik Mei 13, 2022, dari https://ichi.pro/id: https://ichi.pro/id/cara-mengimpor-file-csv-ke-database-mysql-menggunakan-python-133156956822579

SkillPlus. (2021, Januari 15). *Membuat dan Menyimpan DataFrame*. Dipetik Mei 22, 2022, dari skillplus.web.id: https://skillplus.web.id/membuat-dan-menyimpan-dataframe/

Tipler, P. A., & Mosca, G. (2007). *Physics for Scientists and Engineers with Modern Physics.* New York: W. H. Freeman and Company .

Lukito Andriansyah | **lukitoandriansyah45@gmail.com**