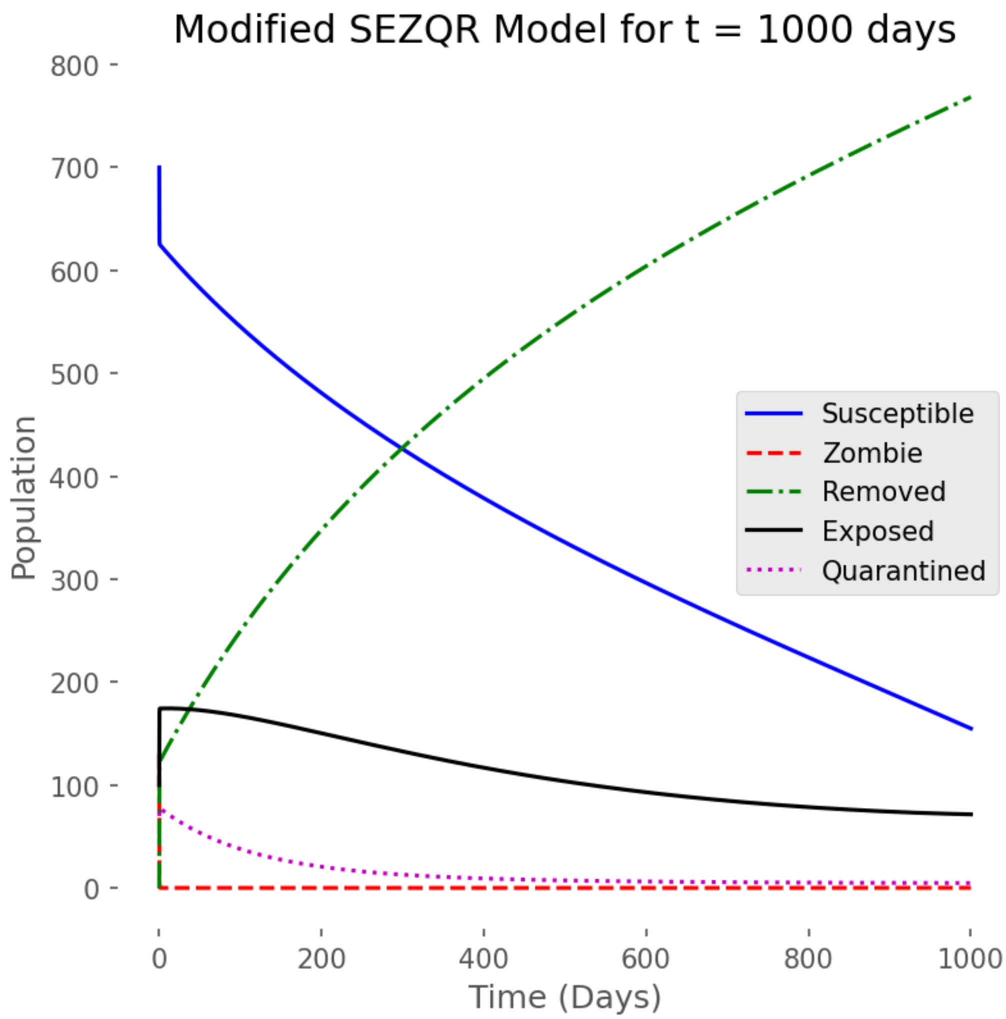
ModifiedSEZQR_Model

September 28, 2021

[64]:
```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-

"""This simple program tries to solve the ODE of the Modified SEZQR model with␣
↪perturbation and create a numerical solution.
   After that, this program draws the solution graph for the Susceptible,␣
↪Exposed, Zombie, Quarantined and Removed
   Population. This simple graph helps us to understand the actual scenario of␣
↪the population
   from the modified SEZQR model after a certain number of days. We can change␣
↪the value of the Perturbation
   Parameter Mu to investigate the change of behaviour and stabality of the␣
↪SZR model.

   Part of MS Thesis at Universität Koblenz-Landau
   Note: this program uses the following libraries-  Numpy, Matplotlib, Scipy
   Python Version 3.7
"""

# MODIFIED SEZQR MODEL

# importing libraries
import numpy as np
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
from matplotlib.pyplot import figure
figure(figsize=(6, 6), dpi=150)

# I'm using this style for a pretier plot, but it's not actually necessary
plt.style.use('ggplot')

# Parameter values
beta = 0.0095
alpha = 0.005
zeta = 0.0001
rho = 0.005
kappa = 0.001
```

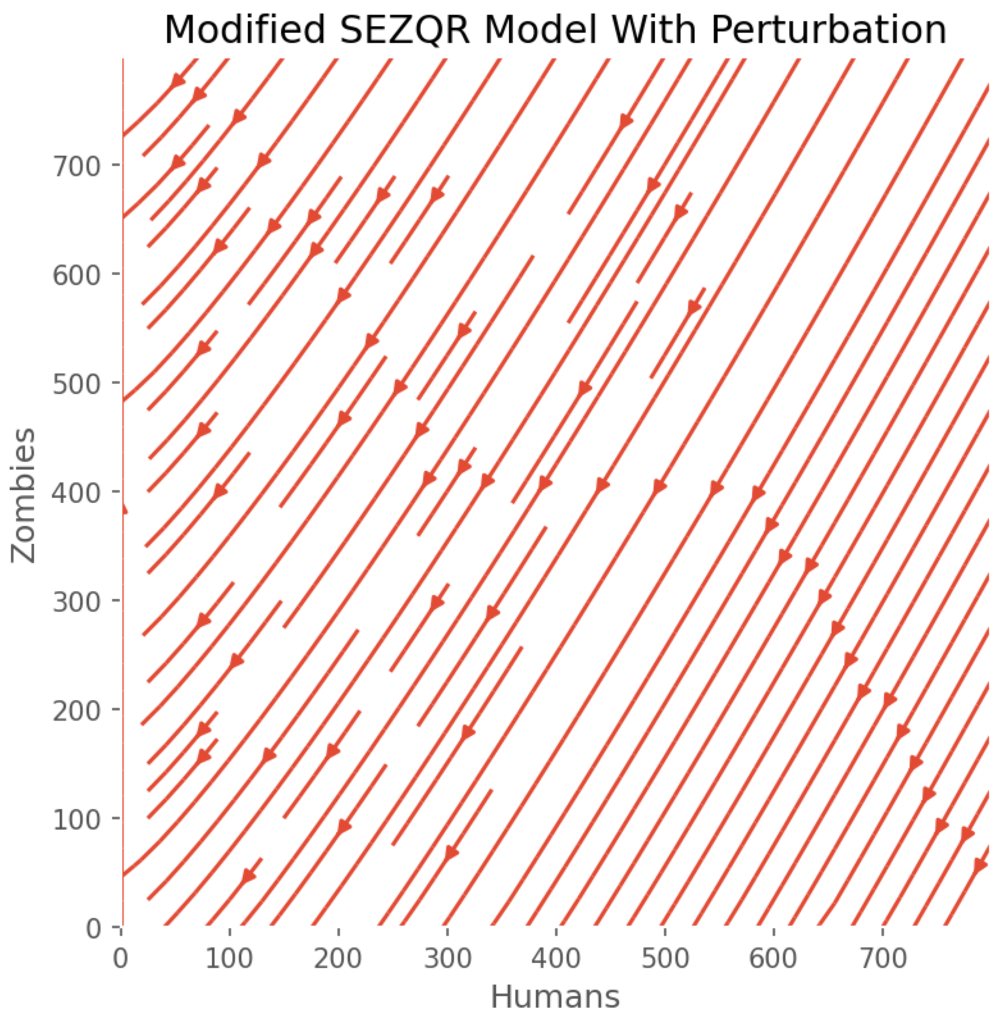
```
sigma = 0.001
gamma = 0.0001
kappa = 0.0001
omega = 0.009
N = 1000  # initial population

# Perturbation Parameter Mu, change this value to see different results
mu = 0.175

# Defining Function of the SEZQR ODE
def model(t, x):
    S = x[0] # Intital Susceptible Population
    E = x[1] # exposed
    Z = x[2] # zombies
    Q = x[3] # quarentined
    dsdt = - beta*S*Z # Susceptible
    dedt = beta*S*Z - (rho + kappa)*E # Exposed
    dzdt = rho*E + zeta*(N - S - E - Z - Q) - sigma*S*Z - alpha*(S**(1 + mu))*Z␣
 ↪+ omega*Q # Zombies
    dqdt = kappa*E + sigma*S*Z - gamma*Q - omega*Q # Quarantined
    dxdt = [dsdt, dedt, dzdt, dqdt]
    return dxdt

# Initial Condition. Susceptible= 700, Exposed= 100, Zombie= 130, Quarantined =␣
 ↪70
x0 = [700, 100, 130, 70]

# Time, as in number of days. Total days= 1000
t = np.array([0, 1000])
tspan = np.linspace(t[0], t[1], 10001)

# Calculating numerical solution of the given ODEs
x = solve_ivp(model, t, x0, t_eval=tspan)

time = x.t
susceptible = x.y[0];
exposed = x.y[1];
zombie = x.y[2]
quarantined  = x.y[3]
removed = (N - susceptible - exposed - zombie - quarantined)

# Plot
plt.plot(time, susceptible, 'b-', label="Susceptible") # Susceptible Population␣
 ↪Graph
plt.plot(time, zombie, 'r--', label="Zombie") # Zombie Population Graph
plt.plot(time, removed, 'g-.', label="Removed") # Recovered Population Graph
plt.plot(time, exposed, 'k', label="Exposed") # Exposed Population graph
```

```
plt.plot(time, quarantined , 'm:', label="Quarantined")
plt.title('Modified SEZQR Model for t = 1000 days')
plt.ylabel('Population')
plt.xlabel('Time (Days)')
plt.legend(loc='best')
ax = plt.gca()
ax.set_facecolor('w')
plt.show()

__author__ = "Md Tariqul Islam"
__version__ = "1.0"
__maintainer__ = "Tariqul"
__email__ = "tariquldipu@uni-koblenz.de"
__status__ = "Final"
```



[63]:
```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-

"""This program plots the Phase Portrait for the Modified SEZQR Model

   Part of MS Thesis at Universität Koblenz-Landau
   Note: this program uses the following libraries-  Numpy, Matplotlib
   Python Version 3.7
"""
# Phase Portrait For Modified SEZQR Model

# importing libraries
import numpy as np
from matplotlib import pyplot as plt
from matplotlib.pyplot import figure
figure(figsize=(6, 6), dpi=150)


# Parameter values
beta = 0.0095
alpha = 0.005
zeta = 0.0001
rho = 0.005
kappa = 0.001
sigma = 0.001
gamma = 0.0001
kappa = 0.0001
omega = 0.009
N = 1000  # initial population

x = np.arange(0, 800, 1)
y = np.arange(0, 800, 1)
S, Z = np.meshgrid(x, y)
p = np.arange(0, 800, 1)
q = np.arange(0, 800, 1)
E, Q = np.meshgrid(p, q)

# range of values for Mu, with an increment of 0.005
mu = 0.175

S_dash = - beta*S*Z #SEZQR model differential equation- Susceptible
#E_dash = beta*S*Z - (rho + kappa)*E # Exposed
Z_dash = rho*E + zeta*(N - S - E - Z - Q) - sigma*S*Z - alpha*(S**(1 + mu))*Z +␣
↪omega*Q #SEZQR model differential equation- Zombie
#Q_dash = kappa*E + sigma*S*Z - gamma*Q - omega*Q # Quarantined
plt.streamplot(S, Z, S_dash, Z_dash, density=1.1) #phase portrait
plt.title("Modified SEZQR Model With Perturbation")
```

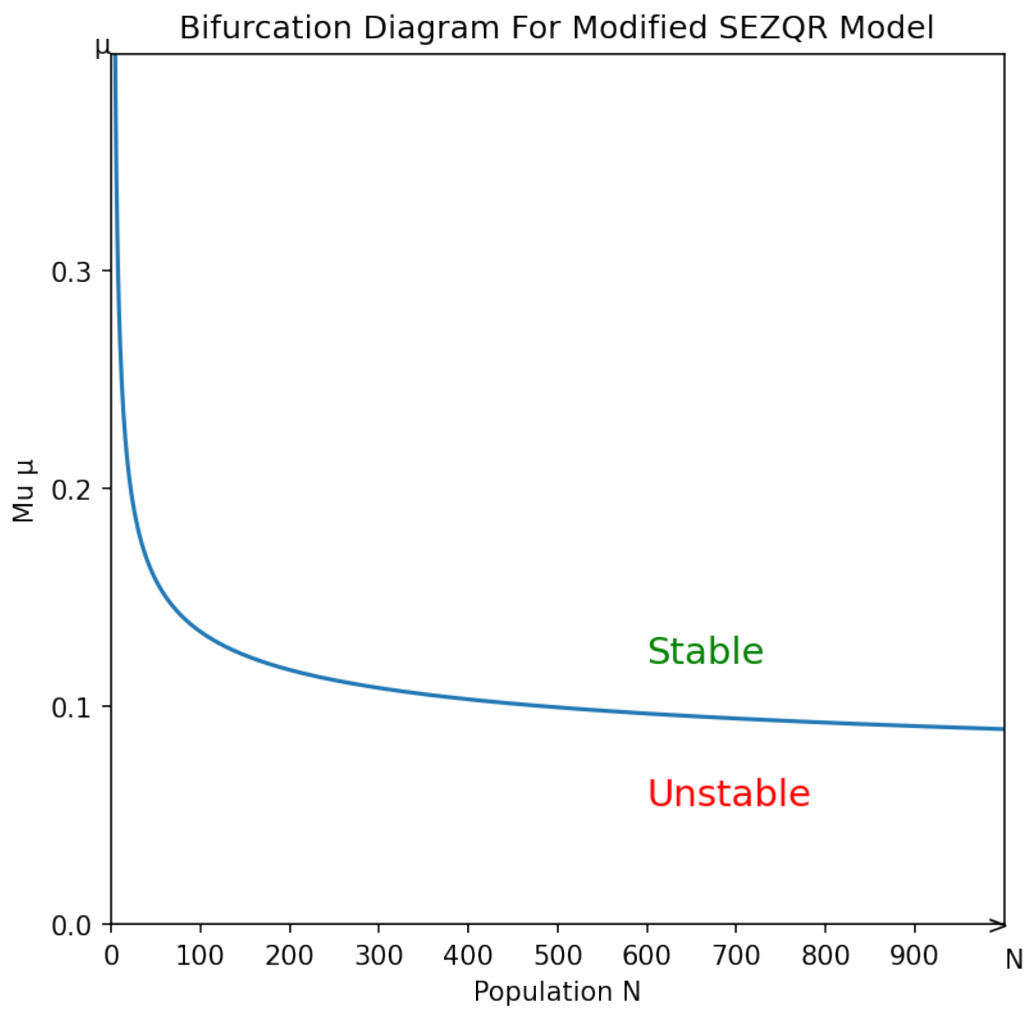
```
plt.axis("scaled")
plt.draw()
plt.xlabel("Humans")
plt.ylabel("Zombies")
ax = plt.gca()
ax.set_facecolor('w')

__author__ = "Md Tariqul Islam"
__version__ = "1.0"
__maintainer__ = "Tariqul"
__email__ = "tariquldipu@uni-koblenz.de"
__status__ = "Final"
```



[2]:
```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
```

```python
"""This program plots the bifurcation diagram for the Modified SEZQR Model

   Part of MS Thesis at Universität Koblenz-Landau
   Note: this program uses the following libraries-  Numpy, Matplotlib
   Python Version 3.7
"""
# Bifurcation Diagram For Modified SEZQR

# MODIFIED SEZQR MODEL

# Import our modules that we are using
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.pyplot import figure
figure(figsize=(6, 6), dpi=150)


beta = 0.0095
alpha = 0.005
zeta = 0.0001
rho = 0.005
kappa = 0.001
sigma = 0.001
gamma = 0.0001
omega = 0.009
N = 1000


# Create the vectors X and Y
x = np.array(range(2,1000))
temp01 = ((zeta-rho)*(gamma+omega)+kappa*(zeta-omega))
temp02 = (rho+kappa)*(sigma*x*(zeta - omega) + (gamma+omega)*(zeta + sigma*x))
y = ((np.log((beta*x*temp01 + temp02 )/(-alpha*(gamma+omega)*(rho+kappa))))/(np.
 ↪log(x))) - 1

# Create the plot
plt.plot(x,y)
plt.ylim([0, 0.4])
plt.xlim([0, 1000])
plt.yticks(np.arange(0, 0.4, 0.1))
plt.xticks(np.arange(0, 1000, 100))
plt.xlabel("Population N")
plt.ylabel("Mu " + u"\u03bc")

plt.text(1000, -0.02, "N")
plt.text(980, -0.004, ">")
```

```python
#plt.text(0, 0.4, ">")
plt.text(-20, 0.4, u"\u03bc")

plt.text(600, 0.12, "Stable",
        fontsize= 14,
        color= 'green')
plt.text(600, 0.055, "Unstable",
        fontsize= 14,
        color= 'red')


# Show the plot
plt.title("Bifurcation Diagram For Modified SEZQR Model")
plt.show()


__author__ = "Md Tariqul Islam"
__version__ = "1.0"
__maintainer__ = "Tariqul"
__email__ = "tariquldipu@uni-koblenz.de"
__status__ = "Final"
```

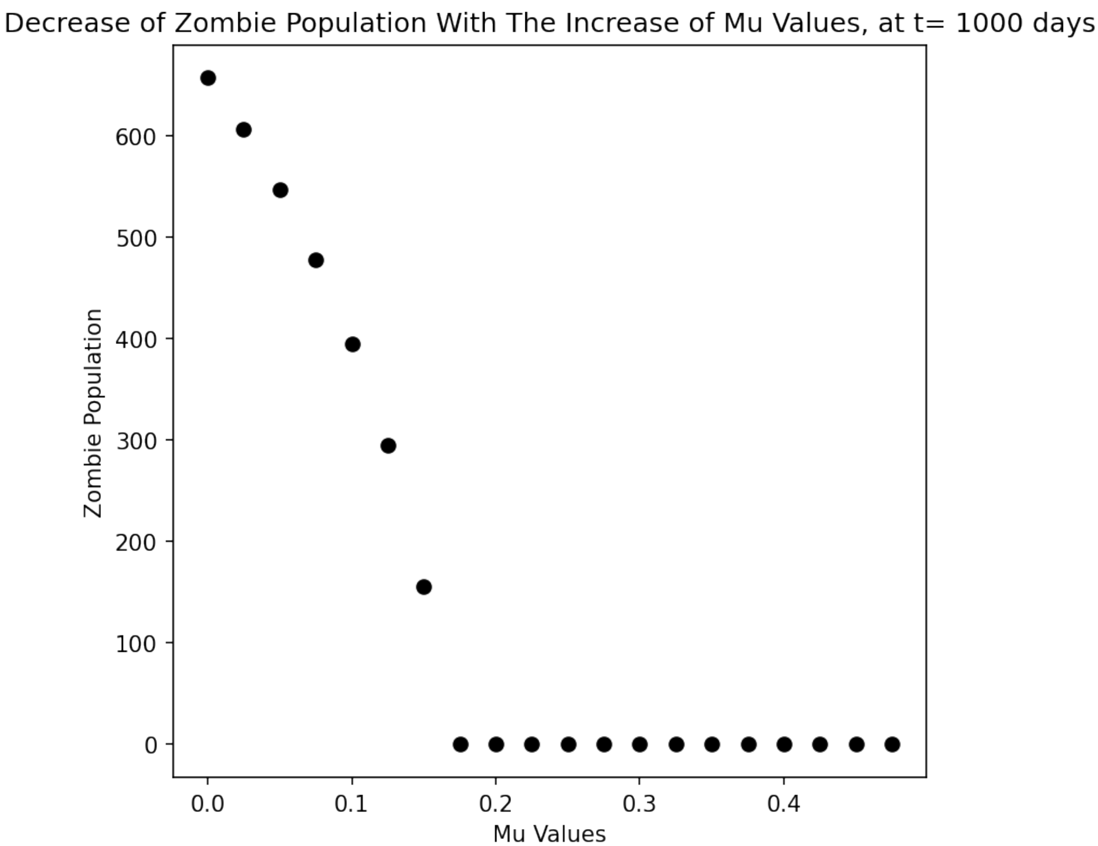


[9]:
```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

# Modified SEZQR Model

"""This simple program tries to show a relation between the number of zombies␣
↪and the perturbation value Mu.
    This program is calculated at the point t = 1000 days.
    It shows that, at t = 1000 days, as the value of the pertubation parameter␣
↪Mu increases, the number of total
    zombies decreases. Thus, establishing the fact that the perturbation␣
↪parameter has a direct impact on the
    zombie population.

   Part of MS Thesis at Universität Koblenz-Landau
```

```
   Note: this program uses the following libraries-  Numpy, Matplotlib, Scipy
   Python Version 3.7
"""

# importing libraries
import numpy as np
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
from matplotlib.pyplot import figure
figure(figsize=(6, 6), dpi=150)
import warnings
warnings.filterwarnings("ignore")

# Generating range of values for Mu, from 0.0 upto 0.5, with an increment of 0.
↪025
mu = np.arange(0.0, 0.5, 0.025)

# Parameter values
beta = 0.0095
alpha = 0.005
zeta = 0.0001
rho = 0.005
kappa = 0.001
sigma = 0.001
gamma = 0.0001
kappa = 0.0001
omega = 0.009
N = 1000  # initial population

def model(t, x):
    S = x[0] # Intital Susceptible Population
    E = x[1] # exposed
    Z = x[2] # zombies
    Q = x[3] # quarentined
    dsdt = - beta*S*Z # Susceptible
    dedt = beta*S*Z - (rho + kappa)*E # Exposed
    dzdt = rho*E + zeta*(N - S - E - Z - Q) - sigma*S*Z - alpha*(S**(1 +␣
↪value))*Z + omega*Q # Zombies
    dqdt = kappa*E + sigma*S*Z - gamma*Q - omega*Q # Quarantined
    dxdt = [dsdt, dedt, dzdt, dqdt]
    return dxdt

# Initial Condition. Susceptible= 700, Exposed= 100, Zombie= 130, Quarantined =␣
↪70
x0 = [700, 100, 130, 70]

# Time, as in number of days. Total days= 1000
```

```
t = np.array([0, 1000])
tspan = np.linspace(t[0], t[1], 1000)

i = 0;

# empty lists
zombie_new = []
recovered_new = []

for value in mu:
    x_new = solve_ivp(model, t, x0, t_eval=tspan)
    zombie = x_new.y[2];
    zombie_new.append(zombie[999])
    #susceptible = x_new.y[0];
    #recovered_new.append((N - susceptible[999] - zombie[999]))

# Drawing Scatter Plot at t = 1000 days position
# Change of zombie numbers with the increase of Mu Values
# When the number of zombies reaches 0 at t= 1000 days, the model reaches␣
 ↪stability at that Mu value
plt.plot(mu, zombie_new, 'o', color='black');
#plt.plot(mu, recovered_new, 'o', color='blue');
plt.ylabel('Zombie Population')
plt.xlabel('Mu Values')
plt.title("Decrease of Zombie Population With The Increase of Mu Values, at t=␣
 ↪1000 days")
ax = plt.gca()
ax.set_facecolor('w')
plt.show
```

[9]: <function matplotlib.pyplot.show(close=None, block=None)>



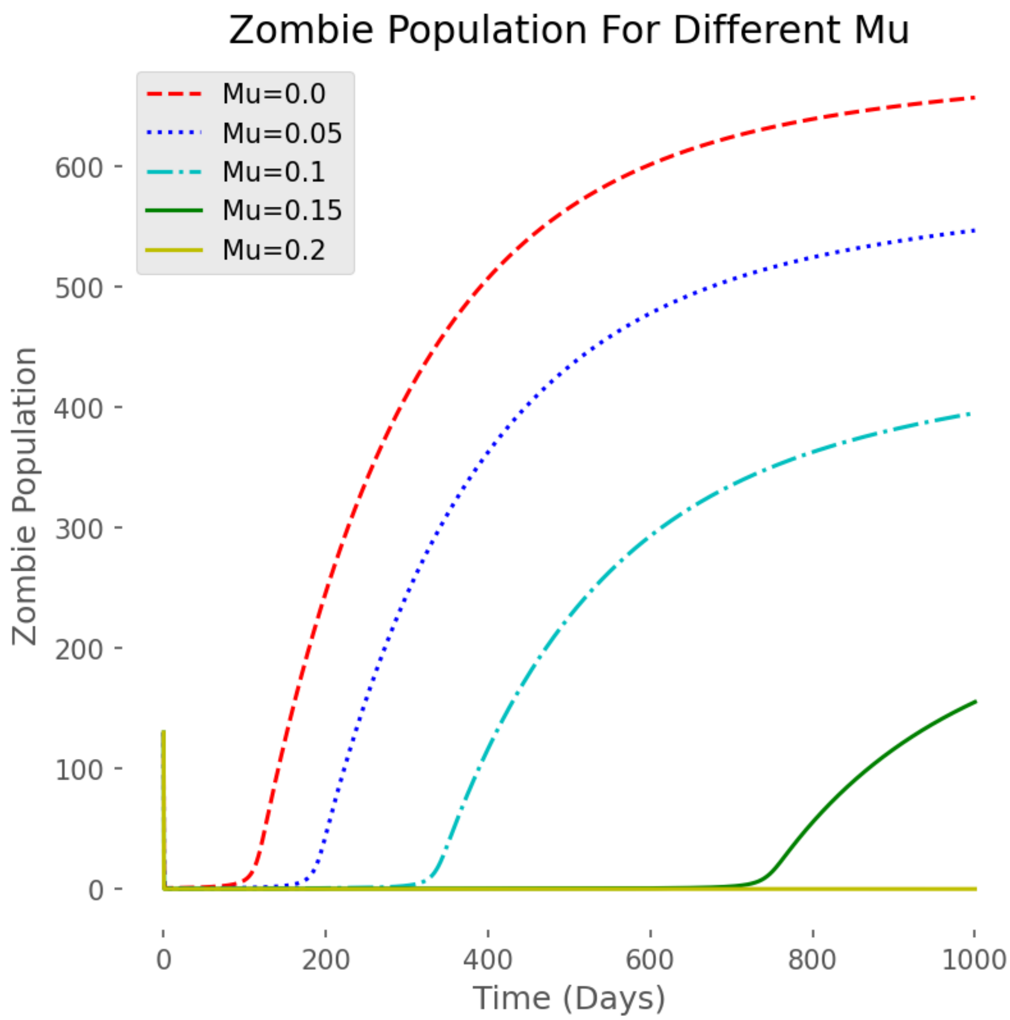
[3]:
```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

"""This simple program tries to solve the ODE of the Modified SEZQR model with␣
↪perturbation
    and create a numerical solution.
    After that, this program draws the solution graph for only the Zombie␣
↪Population with the gradual
    increase of perturbation parameter.
    This simple graph helps us to understand the actual scenario of the zombie␣
↪population
    from the modified SEZQR model with respect to the perturbation parameter.

   Part of MS Thesis at Universität Koblenz-Landau
   Note: this program uses the following libraries-  Numpy, Matplotlib, Scipy
   Python Version 3.7
"""

# MODIFIED SEZQR MODEL
```

```
# importing libraries
import numpy as np
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
from matplotlib.pyplot import figure
figure(figsize=(6, 6), dpi=150)

# I'm using this style for a pretier plot, but it's not actually necessary
plt.style.use('ggplot')

# Parameter values
beta = 0.0095
alpha = 0.005
zeta = 0.0001
rho = 0.005
kappa = 0.001
sigma = 0.001
gamma = 0.0001
kappa = 0.0001
omega = 0.009
N = 1000  # initial population


# Defining Function of the SEZQR ODE
def model(t, x):
    S = x[0] # Intital Susceptible Population
    E = x[1] # exposed
    Z = x[2] # zombies
    Q = x[3] # quarentined
    dsdt = - beta*S*Z # Susceptible
    dedt = beta*S*Z - (rho + kappa)*E # Exposed
    dzdt = rho*E + zeta*(N - S - E - Z - Q) - sigma*S*Z - alpha*(S**(1 + mu))*Z␣
 ↪+ omega*Q # Zombies
    dqdt = kappa*E + sigma*S*Z - gamma*Q - omega*Q # Quarantined
    dxdt = [dsdt, dedt, dzdt, dqdt]
    return dxdt

# Initial Condition. Susceptible= 700, Exposed= 100, Zombie= 130, Quarantined =␣
 ↪70
x0 = [700, 100, 130, 70]

# Time, as in number of days. Total days= 1000
t = np.array([0, 1000])
tspan = np.linspace(t[0], t[1], 10001)

# Calcuclate zombie population for different values of Mu
```

```python
mu = 0.0
x = solve_ivp(model, t, x0, t_eval=tspan)
zombie01 = x.y[2]

mu = 0.05
x = solve_ivp(model, t, x0, t_eval=tspan)
zombie02 = x.y[2]

mu = 0.1
x = solve_ivp(model, t, x0, t_eval=tspan)
zombie03 = x.y[2]

mu = 0.15
x = solve_ivp(model, t, x0, t_eval=tspan)
zombie04 = x.y[2]

mu = 0.2
x = solve_ivp(model, t, x0, t_eval=tspan)
zombie05 = x.y[2]

mu = 0.25
x = solve_ivp(model, t, x0, t_eval=tspan)
zombie06 = x.y[2]

time = x.t
# Plot
plt.plot(time, zombie01, 'r--', label="Mu=0.0")
plt.plot(time, zombie02, 'b:', label="Mu=0.05")
plt.plot(time, zombie03, 'c-.', label="Mu=0.1")
plt.plot(time, zombie04, 'g-', label="Mu=0.15")
plt.plot(time, zombie05, 'y-', label="Mu=0.2")
#plt.plot(time, zombie06, 'k', label="Mu=0.25")

plt.title('Zombie Population For Different Mu')
plt.ylabel('Zombie Population')
plt.xlabel('Time (Days)')
plt.legend(loc='best')
ax = plt.gca()
ax.set_facecolor('w')
plt.show()

__author__ = "Md Tariqul Islam"
__version__ = "1.0"
__maintainer__ = "Tariqul"
__email__ = "tariquldipu@uni-koblenz.de"
__status__ = "Final"
```

```
<ipython-input-3-a03bd618467a>:49: RuntimeWarning: invalid value encountered in
```

```
double_scalars
  dzdt = rho*E + zeta*(N - S - E - Z - Q) - sigma*S*Z - alpha*(S**(1 + mu))*Z +
omega*Q # Zombies
```




```
[ ]:
```