KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

Skaitiniai metodai ir algoritmai (P170B115)

Laboratorinių darbų ataskaita

Atliko:
IFF-1/4 gr. studentas
Mildaras Karvelis
2023 m. gruodžio 22 d.
Priėmė:
Prof. Barauskas Rimantas

KAUNAS 2023

TURINYS

1. Užduotis

1. Žemiau pateikti uždaviniai paprastųjų diferencialinių lygčių sistemų sprendimui. Remdamiesi tame pačiame faile pateiktų fizikinių dėsnių aprašymais, nurodytam variantui sudarykite diferencialinę lygtį arba lygčių sistemą. Lygties ar lygčių sistemos sudarymą paaiškinkite ataskaitoje.
2. Diferencialinę lygtį (arba lygčių sistemą) išspręskite Eulerio ir IV eilės Rungės ir Kutos metodais.
3. Keisdami metodo žingsnį įsitikinkite, kad gavote tikslų sprendinį. Atsakykite į uždavinyje pateiktus klausimus. Tuo pačiu metodu naudojant skirtingus žingsnius gautus sprendinius pavaizduokite viename grafike. Palyginkite metodus tikslumo prasme.
4. Keisdami metodo žingsnį nustatykite didžiausią žingsnį, su kuriuo metodas išlieka stabilus. Tuo pačiu metodu naudojant skirtingus žingsnius gautus sprendinius pavaizduokite viename grafike. Palyginkite metodus stabilumo prasme.
5. Patikrinkite gautą sprendinį su MATLAB standartine funkcija ode45, Python scipy.integrate bibliotekos funkcija solve_ivp ar kitais išoriniais šaltiniais. Tame pačiame grafike turi būti pateikti realizacijose ir naudojant išorinius šaltinius gauti sprendiniai.

Sujungti m_1 ir m_2 masių objektai iššaunami vertikaliai į viršų pradiniu greičiu v_0. Oro pasipriešinimo koeficientas sujungtiems kūnams lygus k_s. Praėjus laikui t_s, objektai pradeda judėti atskirai. Oro pasipriešinimo koeficientai atskirai judantiems objektams atitinkamai yra k_1 ir k_2. Oro pasipriešinimas proporcingas objekto greičio kvadratui. Raskite, kaip kinta objektų greičiai nuo 0 s iki t_{max}. Kada kiekvienas objektas pasieks aukščiausią tašką ir pradės leistis?

1 Lentelė. Uždavinyje naudojami dydžiai.

Varianto numeris	m_1, kg	m_2, kg	v_0, m/s	k_s, kg/m	t_s, s	k_1, kg/m	k_2, kg/m	t_{max}, s
1	0,2	0,4	80	0,015	1	0,02	0,005	15
2	0,15	0,2	70	0,01	2	0,05	0,001	10
3	0,07	0,2	50	0,015	3	0,05	0,01	10
4	0,5	0,25	100	0,002	2	0,02	0,04	15
5	0,6	0,2	200	0,01	2	0,02	0,015	15
6	0,1	0,5	60	0,01	1	0,01	0,005	10
7	0,3	0,3	60	0,005	2	0,05	0,01	10
8	0,05	0,3	100	0,01	3	0,05	0,01	10
9	0,4	0,8	50	0,001	2	0,02	0,02	10
10	0,8	0,8	200	0,01	2	0,02	0,005	15

2. Diferencialinė lygtis

Išreiškiame judančio objekto pagreitį pasinaudoję antrąjį Niutono dėsnį:

$$\vec{F} = m\vec{a}$$

$\vec{F}$ - veikianti jėga, m – masė, $\vec{a}$ - pagreitis

Oro pasipriešinimo jėga yra proporcinga greičio kvadratui:

$$F_{pasipriešinimo} = -kv^2$$

k - oro pasipriešinimo koeficientas, v – greitis

Iki objektų atsiskyrimo jėgų pusiausvyros lygtis :

$$m_1 a_1 = -k_1 v_1^2$$

$$m_2 a_2 = -k_2 v_2^2$$

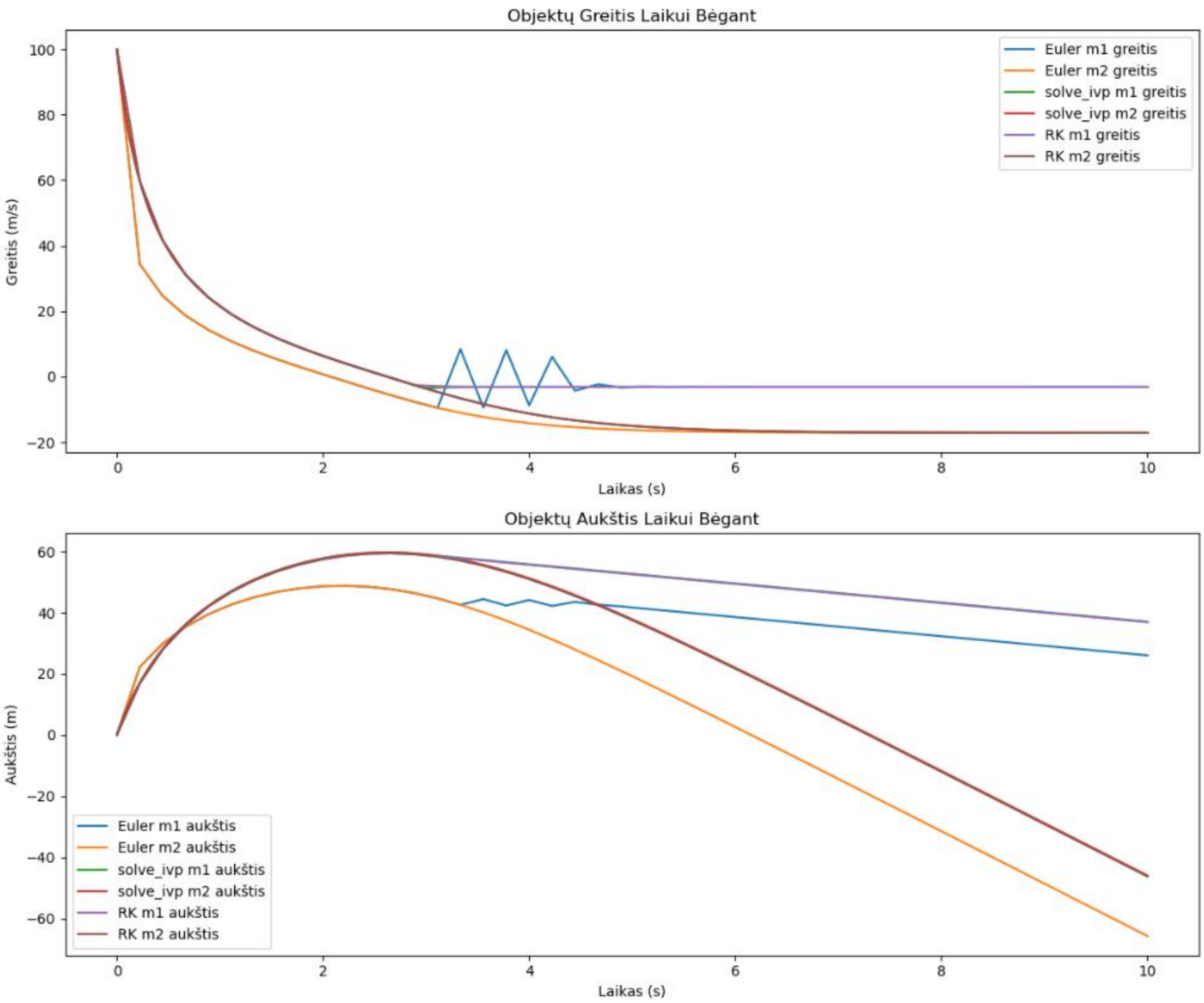
Sudarome diferencialinę lygtį:

$$\frac{dv}{dt} = -g - \left(\frac{kv^2 sgn(v)}{m}\right)$$

3. Sprendimas

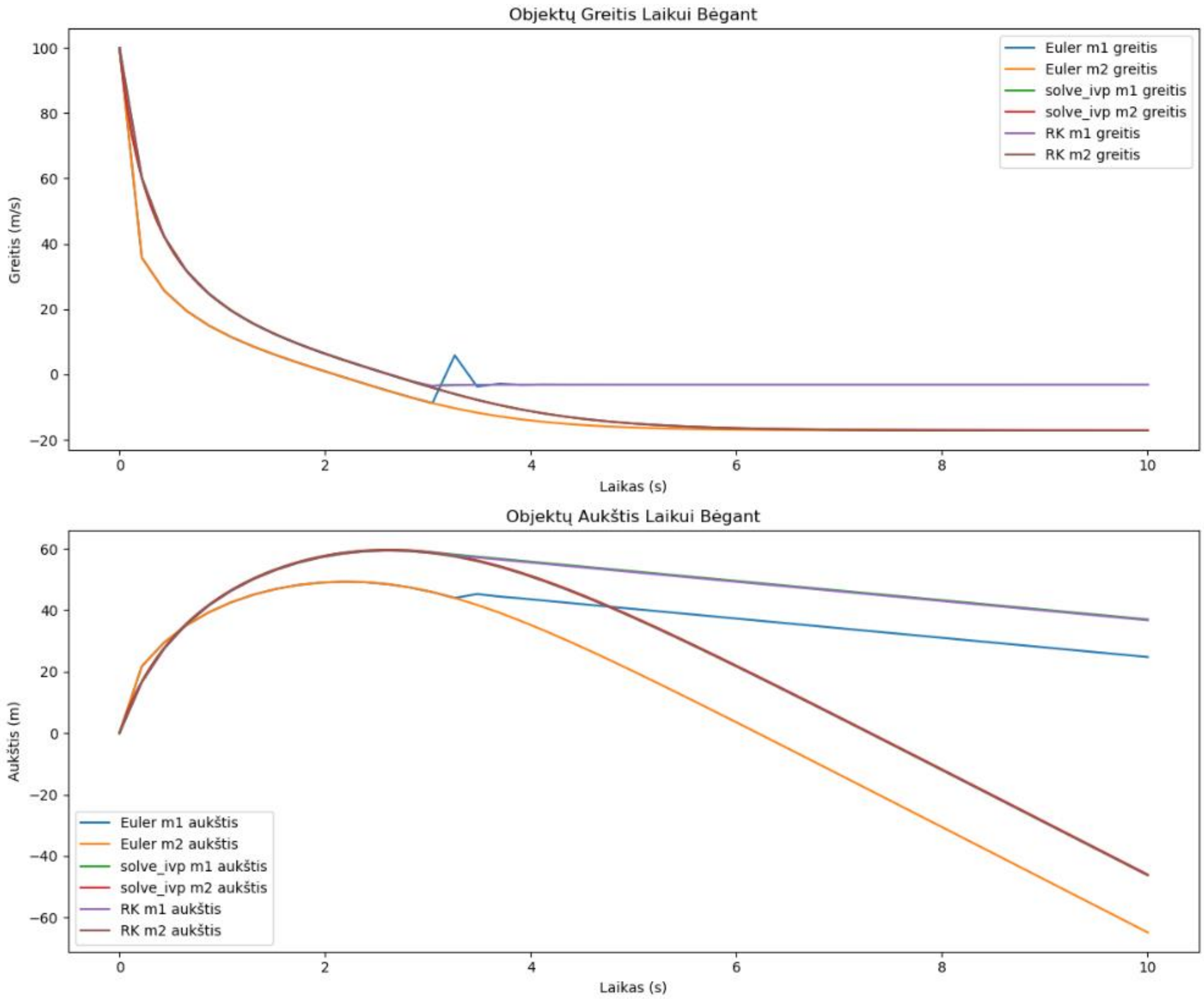
3.1. Žingsnis-46

Kūnas m1 (Euler) pasiekia max aukštį: 2.0 s
Kūnas m2 (Euler) pasiekia max aukštį: 2.0 s
Kūnas m1 (RK) pasiekia max aukštį: 2.4444444444444444 s
Kūnas m2 (RK) pasiekia max aukštį: 2.4444444444444444 s
Kūnas m1 (solve_ivp) pasiekia max aukštį: 2.610441767068273 s
Kūnas m2 (solve_ivp) pasiekia max aukštį: 2.610441767068273 s



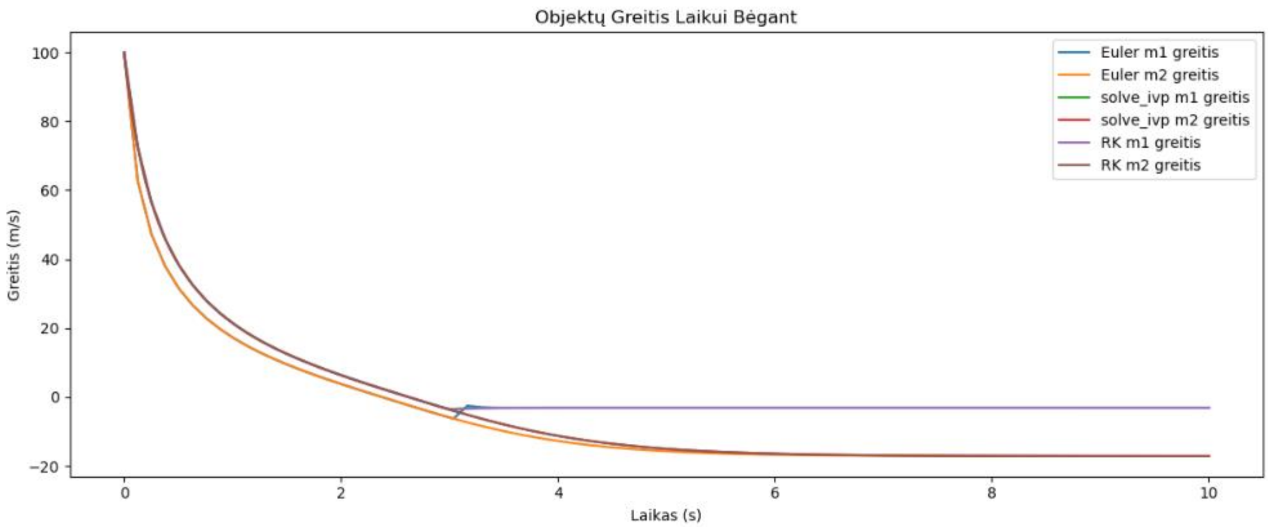
3.2. Žingsnis-47

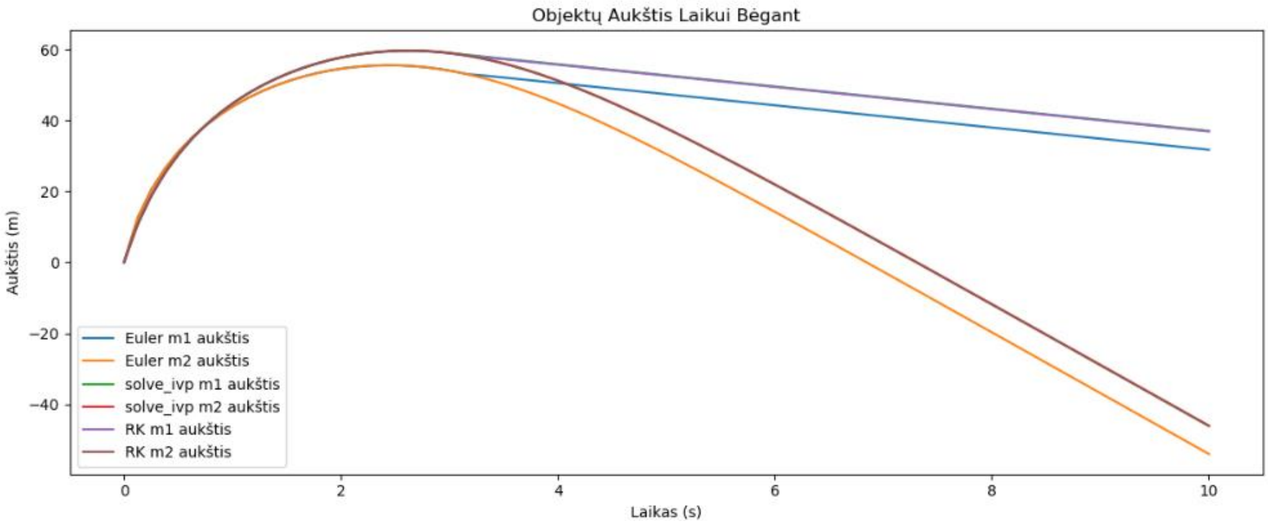
Kūnas m1 (Euler) pasiekia max aukštį: 1.9565217391304348 s
Kūnas m2 (Euler) pasiekia max aukštį: 1.9565217391304348 s
Kūnas m1 (RK) pasiekia max aukštį: 2.608695652173913 s
Kūnas m2 (RK) pasiekia max aukštį: 2.608695652173913 s
Kūnas m1 (solve_ivp) pasiekia max aukštį: 2.610441767068273 s
Kūnas m2 (solve_ivp) pasiekia max aukštį: 2.610441767068273 s



3.3. Žingsnis-80

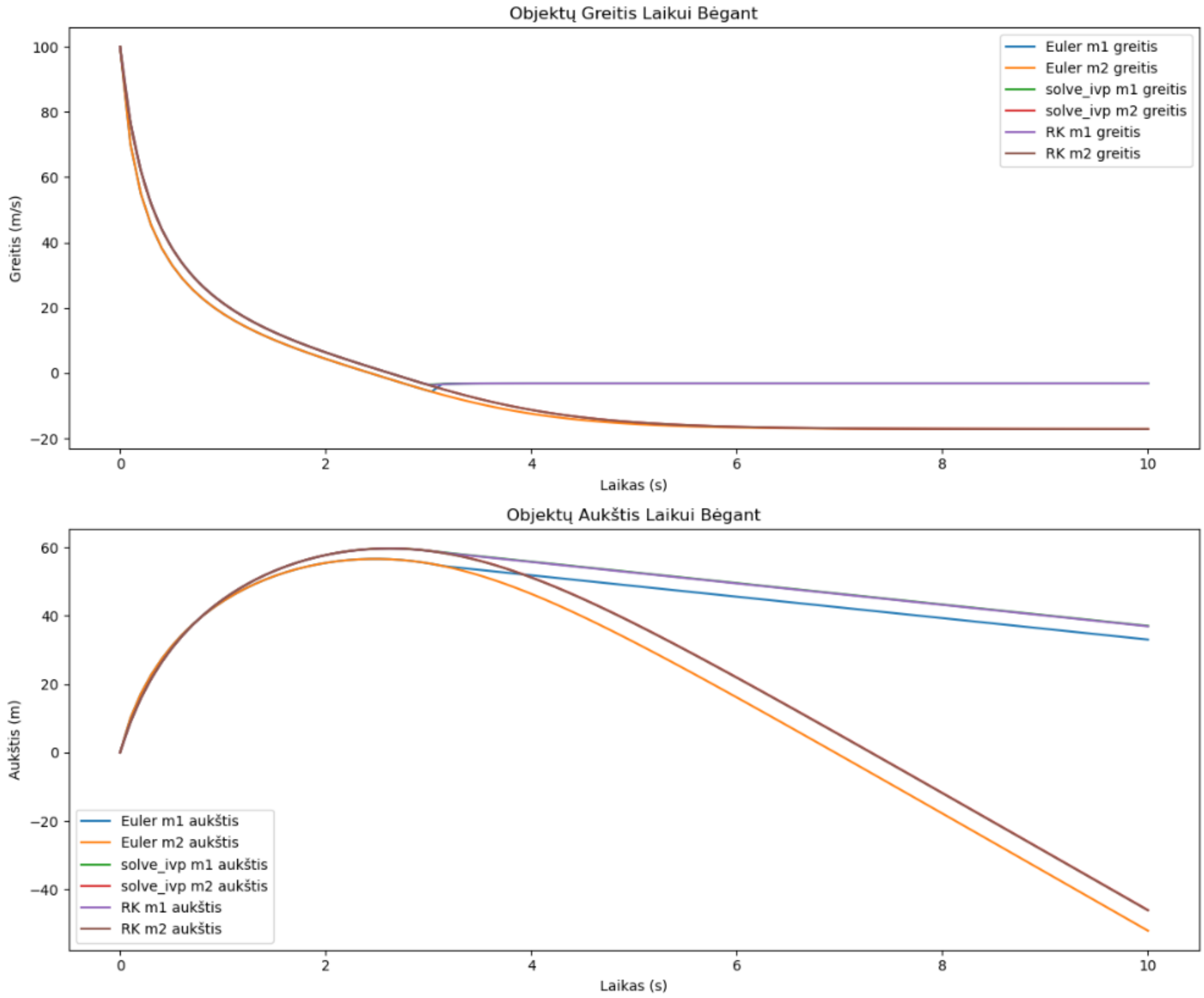
Kūnas m1 (Euler) pasiekia max aukštį: 2.278481012658228 s
Kūnas m2 (Euler) pasiekia max aukštį: 2.278481012658228 s
Kūnas m1 (RK) pasiekia max aukštį: 2.5316455696202533 s
Kūnas m2 (RK) pasiekia max aukštį: 2.5316455696202533 s
Kūnas m1 (solve_ivp) pasiekia max aukštį: 2.610441767068273 s
Kūnas m2 (solve_ivp) pasiekia max aukštį: 2.610441767068273 s





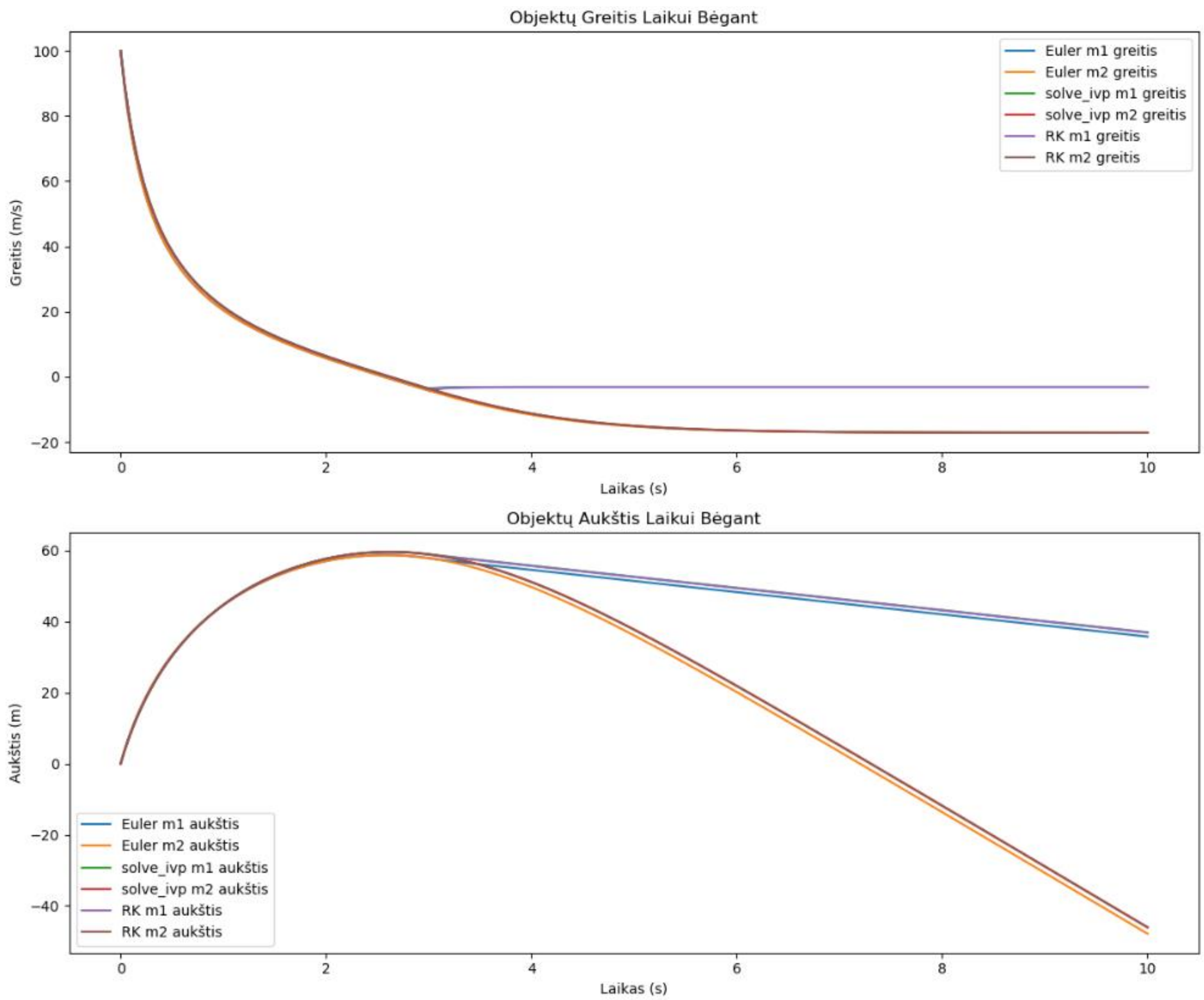
3.4. Žingsnis-100

Kūnas m1 (Euler) pasiekia max aukštį: 2.4242424242424243 s
Kūnas m2 (Euler) pasiekia max aukštį: 2.4242424242424243 s
Kūnas m1 (RK) pasiekia max aukštį: 2.525252525252525 s
Kūnas m2 (RK) pasiekia max aukštį: 2.525252525252525 s
Kūnas m1 (solve_ivp) pasiekia max aukštį: 2.610441767068273 s
Kūnas m2 (solve_ivp) pasiekia max aukštį: 2.610441767068273 s



3.5. Žingsnis-300

Kūnas m1 (Euler) pasiekia max aukštį: 2.5418060200668897 s
Kūnas m2 (Euler) pasiekia max aukštį: 2.5418060200668897 s
Kūnas m1 (RK) pasiekia max aukštį: 2.608695652173913 s
Kūnas m2 (RK) pasiekia max aukštį: 2.608695652173913 s
Kūnas m1 (solve_ivp) pasiekia max aukštį: 2.610441767068273 s
Kūnas m2 (solve_ivp) pasiekia max aukštį: 2.610441767068273 s



4. Programos kodas

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import solve_ivp

# Duotos konstantos
m1 = 0.05  # kg
m2 = 0.3  # kg
v0 = 100  # m/s
ks = 0.01  # kg/m sujungtiems objektams
ts = 3  # s, laikas po kurio objektai atskiriami
k1 = 0.05  # kg/m m1 masėi
k2 = 0.01  # kg/m m2 masėi
tmax = 10  # s, bendras laiko tarpas

# Gravitacija
g = 9.8  # m/s^2

# Diferencialinės lygties funkcija
def diferencialine_lygtis(t, Y):
    v1, v2, h1, h2 = Y
    f = np.zeros(4)
    if t < ts:
        f[0] = f[1] = -g - (ks * v1**2 * np.sign(v1)) / (m1 + m2)
    else:
```

```
        f[0] = -g - (k1 * v1**2 * np.sign(v1)) / m1
        f[1] = -g - (k2 * v2**2 * np.sign(v2)) / m2
    f[2] = v1
    f[3] = v2
    return f

# Eulerio metodo funkcija
def Eulerio_Zingsnis(Y, diferencialine_lygtis, t, dt):
    return Y + diferencialine_lygtis(t, Y) * dt

# IV eilės Rungės-Kutos metodo funkcija
def IV_RK(Y, diferencialine_lygtis, t, dt):
    k1 = diferencialine_lygtis(t, Y)
    k2 = diferencialine_lygtis(t + dt / 2, Y + dt / 2 * k1)
    k3 = diferencialine_lygtis(t + dt / 2, Y + dt / 2 * k2)
    k4 = diferencialine_lygtis(t + dt, Y + dt * k3)
    return Y + dt / 6 * (k1 + 2 * k2 + 2 * k3 + k4)

# Integracija naudojant scipy solve_ivp
sp_ivp_sprendinys = solve_ivp(diferencialine_lygtis, [0, tmax], [v0, v0, 0, 0], t_eval=np.linspace(0, tmax, 250), method='RK45')

# Pradinės sąlygos ir parametrai
Y0 = [v0, v0, 0, 0]
NSteps = 300

t_eval = np.linspace(0, tmax, NSteps)
dt = t_eval[1] - t_eval[0]
Y_euler = np.zeros((4, NSteps))
Y_rk = np.zeros((4, NSteps))
Y_euler[:, 0] = Y0
Y_rk[:, 0] = Y0

# Integracijos metodų taikymas
for i in range(NSteps - 1):
    Y_euler[:, i + 1] = Eulerio_Zingsnis(Y_euler[:, i], diferencialine_lygtis, t_eval[i], dt)
    Y_rk[:, i + 1] = IV_RK(Y_rk[:, i], diferencialine_lygtis, t_eval[i], dt)



# Randame laiką, kada aukštis didžiausias
def rasti_didziausio_aukscio_laika(greiciai, laikai):
    didziausio_aukscio_indeksai = np.where(np.diff(np.sign(greiciai)))[0]
    if len(didziausio_aukscio_indeksai) > 0:
        return laikai[didziausio_aukscio_indeksai[0]]
    else:
        return None

# Taikome funkciją kiekvienam metodui
m1_max_aukstis_laikas_euler = rasti_didziausio_aukscio_laika(Y_euler[0, :], t_eval)
m2_max_aukstis_laikas_euler = rasti_didziausio_aukscio_laika(Y_euler[1, :], t_eval)

print(f"Kūnas m1 (Euler) pasiekia max aukštį: {m1_max_aukstis_laikas_euler} s")
print(f"Kūnas m2 (Euler) pasiekia max aukštį: {m2_max_aukstis_laikas_euler} s")
m1_max_aukstis_laikas_rk = rasti_didziausio_aukscio_laika(Y_rk[0, :], t_eval)
m2_max_aukstis_laikas_rk = rasti_didziausio_aukscio_laika(Y_rk[1, :], t_eval)
print(f"Kūnas m1 (RK) pasiekia max aukštį: {m1_max_aukstis_laikas_rk} s")
print(f"Kūnas m2 (RK) pasiekia max aukštį: {m2_max_aukstis_laikas_rk} s")
m1_max_aukstis_laikas_solve_ivp = rasti_didziausio_aukscio_laika(sp_ivp_sprendinys.y[0, :], sp_ivp_sprendinys.t)
```

```
m2_max_aukstis_laikas_solve_ivp = rasti_didziausio_aukscio_laika(sp_ivp_sprendinys.y[1, :], sp_ivp_sprendinys.t)
print(f"Kūnas m1 (solve_ivp) pasiekia max aukštį: {m1_max_aukstis_laikas_solve_ivp} s")
print(f"Kūnas m2 (solve_ivp) pasiekia max aukštį: {m2_max_aukstis_laikas_solve_ivp} s")
# Grafikų braižymas
plt.figure(figsize=(12, 10))
# Greičio grafikai
plt.subplot(2, 1, 1)
plt.plot(t_eval, Y_euler[0], label='Euler m1 greitis')
plt.plot(t_eval, Y_euler[1], label='Euler m2 greitis')
plt.plot(sp_ivp_sprendinys.t, sp_ivp_sprendinys.y[0], label='solve_ivp m1 greitis')
plt.plot(sp_ivp_sprendinys.t, sp_ivp_sprendinys.y[1], label='solve_ivp m2 greitis')
plt.plot(t_eval, Y_rk[0], label='RK m1 greitis')
plt.plot(t_eval, Y_rk[1], label='RK m2 greitis')
plt.xlabel('Laikas (s)')
plt.ylabel('Greitis (m/s)')
plt.title('Objektų Greitis Laikui Bėgant')
plt.legend()
# Aukščio grafikai
plt.subplot(2, 1, 2)
plt.plot(t_eval, Y_euler[2], label='Euler m1 aukštis')
plt.plot(t_eval, Y_euler[3], label='Euler m2 aukštis')
plt.plot(sp_ivp_sprendinys.t, sp_ivp_sprendinys.y[2], label='solve_ivp m1 aukštis')
plt.plot(sp_ivp_sprendinys.t, sp_ivp_sprendinys.y[3], label='solve_ivp m2 aukštis')
plt.plot(t_eval, Y_rk[2], label='RK m1 aukštis')
plt.plot(t_eval, Y_rk[3], label='RK m2 aukštis')
plt.xlabel('Laikas (s)')
plt.ylabel('Aukštis (m)')
plt.title('Objektų Aukštis Laikui Bėgant')
plt.legend()
plt.tight_layout()
plt.show()
```