KAUNO TECHNOLOGIJOS UNIVERSITETAS

Skaitiniai metodai ir algoritmai (P170B115)

Projektas

Ketvirta užduotis

7 variantas

Atliko:

IFF-8/12 gr. studentas

Jokūbas Akramas

2020 m. gruodžio 14 d.

Priėmė:

lekt. Darius Naujokaitis

KAUNAS 2020

Turinys

1. PDL sprendimas

1.1. Užduotis

$m1$ masės parašiutininkas su $m2$ masės įranga iššoka iš lėktuvo, kuris skrenda aukštyje $h0$. Po tg laisvo kritimo parašiutas išskleidžiamas. Oro pasipriešinimo koeficientas laisvo kritimo metu lygus $k1$, o išskleidus parašiutą - $k2$. Tariama, kad paliekant lėktuvą parašiutininko greitis lygus 0 m/s, o oro pasipriešinimas proporcingas parašiutininko greičio kvadratui. Raskite, kaip kinta parašiutininko greitis nuo 0 s iki nusileidimo. Kada ir kokiu greičiu parašiutininkas pasiekia žemę? Kokiame aukštyje išskleidžiamas parašiutas?

Varianto numeris	m1, kg	m2, kg	h0, m	tg, s	k1, kg/m	k2, kg/m
7	70	15	4000	40	0,1	5

1.2. Diferencialinės lygties sudarymas

Pagal užduoties sąlygą kūnas juda su pagreičiu, todėl remsiuosi antruoju Niutono dėsniu, kuris teigia, kad pagreitis $\vec{a}$, kuriuo juda kūnas yra tiesiogiai proporcingas kūną veikiančiai jėgai $\vec{F}_A$ ir atvirkščiai proporcingas to kūno masei.

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

Kūną veikianti atstojanti jėga $\vec{F}_A$ yra lygi dviejų kūną veikiančių jėgų sumai – oro pasipriešinimo ir svorio.

$$\vec{F}_A = \vec{F}_{\text{oro pasipriešinimo}} + \vec{F}_{\text{svorio}}$$

Svorio jėga visada bus lygi masės ir laisvojo kitimo pagreičio sandaugai, nukreipta žemės branduolio link. Pasirinkę koordinatės ašį nuo dangaus link žemės, galime užrašyti kūną veikiančią svorio jėgą:

$$\vec{F}_{\text{svorio}} = \mathrm{mg}$$

Oro pasipriešinimo jėga priklausys nuo kūno judėjimo krypties oro atžvilgiu (priešingai nukreiptai) ir proporcinga greičio kvadrato ir pasipriešinimo koeficiento sandaugai. Kūną veikiančią oro pasipriešinimo jėgą užrašome taip:

$$\vec{F}_{\text{oro pasipriešinimo}} = -\mathrm{k}\vec{v}^{2}$$

(kur k – oro pasipriešinimo koeficientas, priklausantis nuo laiko t (kai t < tg, tai k = k1; kai t >= tg, tai k = k2))

Kadangi užduotyje reikalingas tik vertikalus greitis, greičio vektorių galime išreikšti kaip reikšmę:

$$\vec{F}_{\text{oro pasipriešinimo}} = -\mathrm{k}v^2$$

Sudėję abi jėgas, gausime atstojančią jėgą:

$$\vec{F}_A = \mathrm{mg} - \mathrm{k}v^2$$

Išreiškus jėgą remiantis antruoju Niutono dėsniu gauname:

$$m\vec{a} = \mathrm{mg} - \mathrm{k}v^2$$

Pertvarkius lygtį gauname:

$$\vec{a} = g - \frac{kv^2}{m}$$

Iš teorijos žinodami, kad kelio išvestinė pagal laiką yra greitis, o greičio – pagreitis, galime išsireikšti lygčių sistemą:

$$\begin{cases} \dfrac{dh}{dt} = v \\ \dfrac{dv}{dt} = g - \dfrac{kv^2}{m}, \qquad k = \begin{cases} k1, & kai\ t < tg \\ k2, & kai\ t \geq tg \end{cases} \end{cases}$$

1.3. Sprendimo metodų sudarymas

Greičio pokyčio laike skaičiavimo metodo (f1) bei Eulerio (Eulerio) ir 4 eilės Rungės ir Kutos (RK) metodų realizacija.

```
private double f(double v, double k)
{
    int m1 = 70, m2 = 15;
    double g = 9.8, m = m1 + m2;
    //---
    return g - ((k * Math.Pow(v, 2)) / m);
}
private double Eulerio(double v, double step, double k)
{
    return v + step * f(v, k);
}
private double RK(double v, double step, double k)
{
    double v1 = v + (step / 2) * f(v, k);
    double v2 = v + (step / 2) * f(v1, k);
    double v3 = v + step * f(v2, k);
    double v4 = v + (step / 6) * (f(v, k) + 2 * f(v1, k) + 2 * f(v2, k) + f(v3, k));
    return v4;
}
```

Uždavinio sprendimo metodas.

```
private void Sprendimas(double step, ref double[] t_array, ref double[] h_array, ref double[] v_array, out int it_iskleidimas, out int it_nusileidimas)
{
    double k1 = 0.1, k2 = 5, h = 4000, tg = 40, v = 0, k = k1, t = 0;
    //---
    List<double> h_list = new List<double>();
    List<double> v_list = new List<double>();
    List<double> t_list = new List<double>();
    //---
    it_iskleidimas = 0;
    it_nusileidimas = 0;
    //---
    h_list.Add(h);
    v_list.Add(v);
    t_list.Add(t);
    //---
    for (int i = 0; i < 5000; i++)
    {
        if (t >= tg && k == k1)
        {
            k = k2;
            it_iskleidimas = i;
        }
        //---
        if (radioButton3.Checked) v = Eulerio(v, step, k);
        else if (radioButton4.Checked) v = RK(v, step, k);
        h -= step * v;
        t += step;
        //---
        if (h > 0)
        {
            h_list.Add(h);
            v_list.Add(v);
            t_list.Add(t);
        }
        else
        {
            it_nusileidimas = i;
            t_array = t_list.ToArray();
            h_array = h_list.ToArray();
            v_array = v_list.ToArray();
            break;
        }
    }
}
```

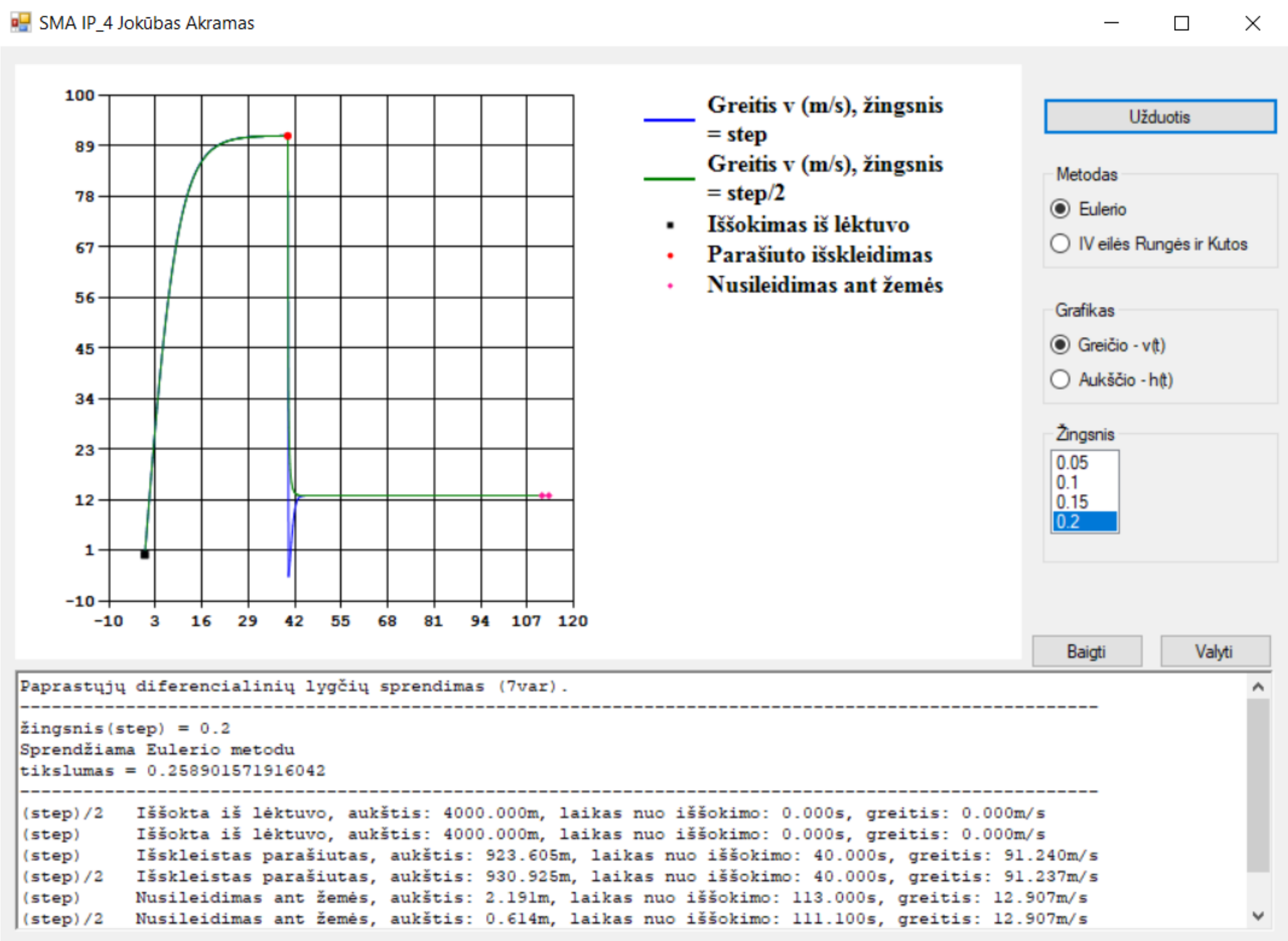
Tikslumo skaičiavimo metodas, kai lyginamos dvi kreivės, o antros kreivės žingsnis dvigubai trumpesnis.

```
private double tikslumas()
{
    int pilnas = t_step_pilnas.Length;
    int pusiau = t_step_pusiau.Length;
    int it_size = (pilnas > pusiau/2) ? pusiau/2 : pilnas;
    double suma = 0;
    for (int i = 0; i < it_size; i++)
    {
        suma += (v_step_pilnas[i] - v_step_pusiau[i * 2]);
        System.Diagnostics.Debug.WriteLine(string.Format("reiksme = {2}, (t) = pilnas: {0}, pusiau: {1}\n", t_step_pilnas[i], t_step_pusiau[i*2], suma));
    }
    return Math.Abs(suma / it_size);
}
```

1.4. Rezultatai

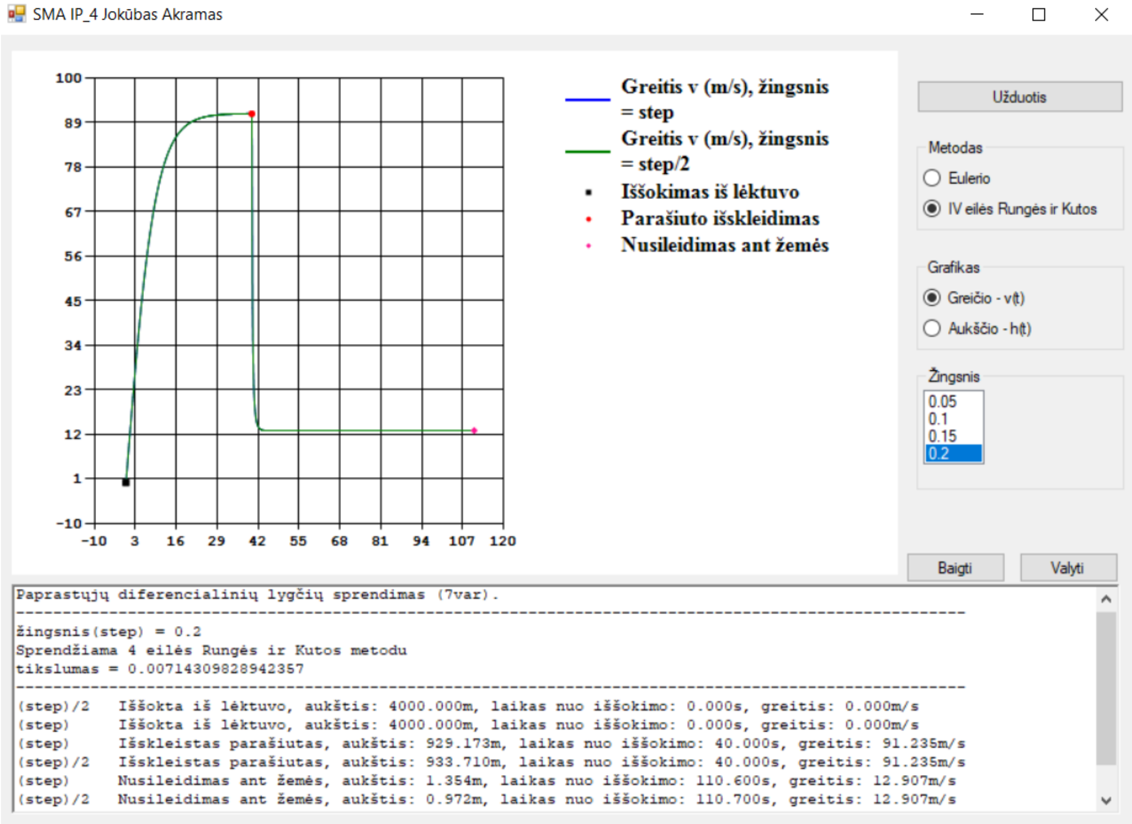
Toliau pateikiami skirtingų metodų, skirtingais žingsniais, rezultatai; tikslumo palyginimai; užduoties klausimų atsakymai.

PDL sprendimas Eulerio metodu su žingsniu 0.2. Greičio grafikas (pav. 1):



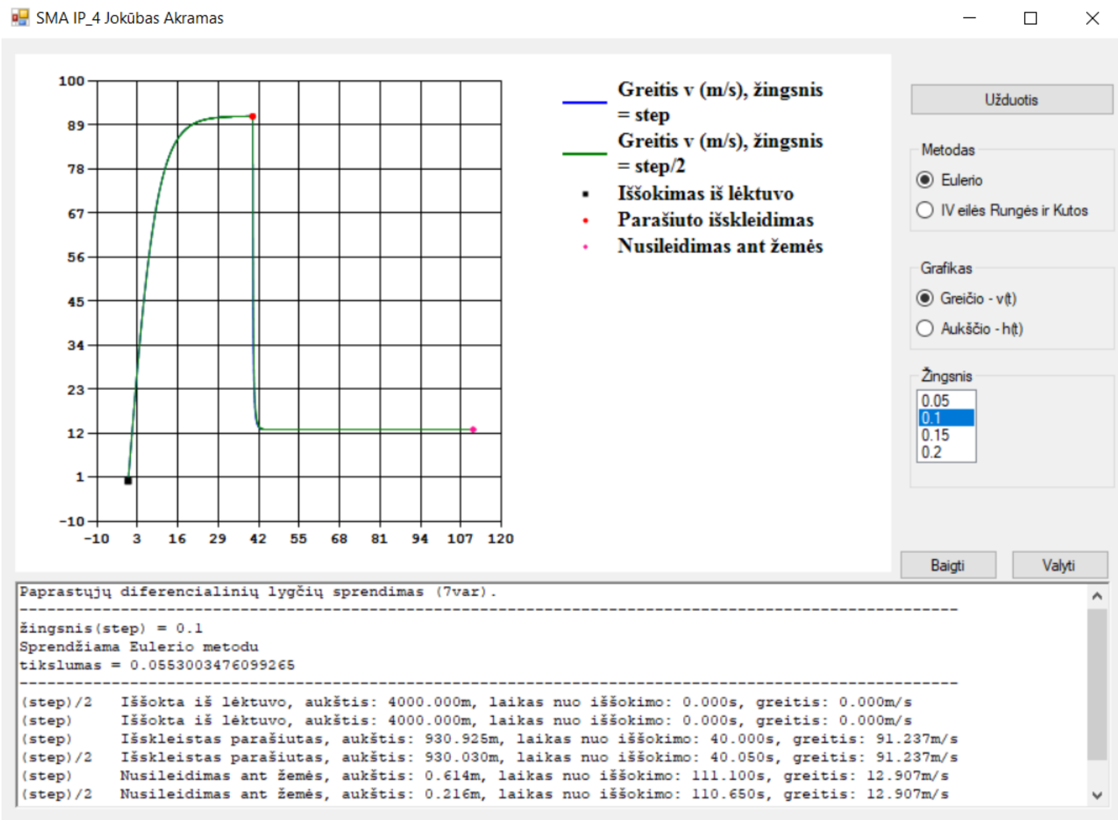
pav. 1 PDL sprendimas Eulerio metodu, greičio grafikas

PDL sprendimas 4 eilės Rungės ir Kutos metodu su žingsniu 0.2. Greičio grafikas (pav. 2):



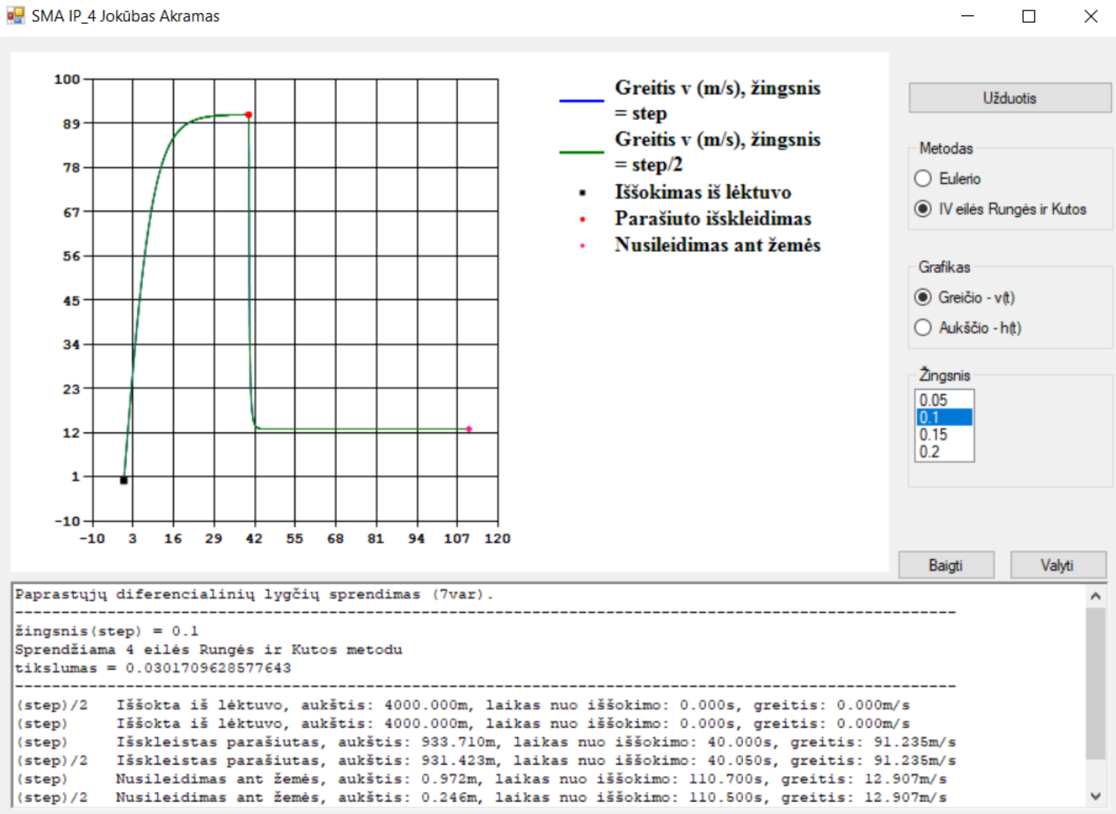
pav. 2 PDL sprendimas 4 eilės Rungės ir Kutos metodu, greičio grafikas

PDL sprendimas Eulerio metodu su žingsniu 0.1. Greičio grafikas (pav. 3):



pav. 3 PDL sprendimas Eulerio metodu, greičio grafikas

PDL sprendimas 4 eilės Rungės ir Kutos metodu su žingsniu 0.1. Greičio grafikas (pav. 4):



pav. 4 PDL sprendimas 4 eilės Rungės ir Kutos metodu, greičio grafikas

Lentelėje pateikti tikslumų įverčiai skaičiuojant abejais metodais, keturiais skirtingais žingsniais (lentelė 1).

lentelė 1 Tikslumų įverčiai

Metodas / Žingsnis	**Eulerio**	**4 eilės Rungės ir Kutos**
0.05	0.0098352758519859	0.0190641682998929
0.1	0.0553003476099265	0.0301709628577643
0.15	0.0729651570562505	0.00078351832097184
0.2	0.258901571916042	0.00714309828942357
Vidurkis	**0.0993**	**0.0143**

Remiantis praktiniais bandymais, 4 eilės Rungės ir Kutos metodas vidutiniškai beveik 7 kartais buvo tikslesnis už Eulerio metodą.

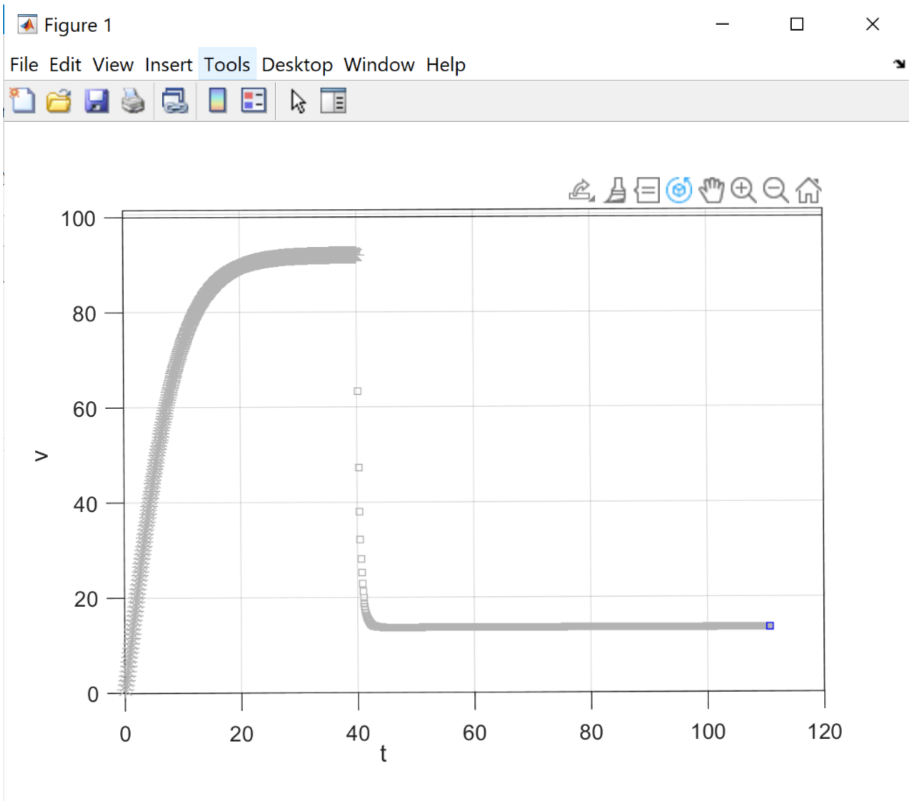
Remiantis programos skaičiavimais:

Kada ir kokiu greičiu parašiutininkas pasiekia žemę? – **Ats.: praėjus ~110s nuo iššokimo iš lėktuvo, ~12.9 m/s greičiu.**

Kokiame aukštyje išskleidžiamas parašiutas? – **Ats.: ~930m**

1.5. Tikrinimas

Toliau pateikiamas pagrindinio uždavinio sprendinio – greičio kitimo grafiko tikrinimas MATLAB standartine funkcija ode45 (pav. 5).



pav. 5 MATLAB tikrinimas

Tikrinimo kodas (MATLAB).

```
function RK4_metodas_parasiutas

clc, clear all,
close all

% sistemos parametrai:
m=85, g=9.81, H=4000, Tp=40;  %mase, pradinis aukstis
v0=0 , %pradinis greitis
c1=0.1 %Ns/m  pasipriesinimo koeff be parasiuto
c2=5 %Ns/m  horizintalus pasipriesinimo koeff su parasiutu
%cv=5 %Ns/m  vertikalus pasipriesinimo koeff su parasiutu
%cp=[ch;ch;cv];
iskleidimas = 0
nusileidimas = 0
h0=H %pradine padetis

color='b';

tmax=200;           % sprendimo intervalo pabaiga
dt=0.1;
figure(1),set(gcf,'Color','w'); hold on, grid on, box on
axis equal;
xlabel('t');ylabel('v');axis equal
%zh=0:H/20:H;izh=length(zh);
axis([0 120 0 100]);%xrng=xlim;yrng=ylim;
%fill3([xrng(1),xrng(2),xrng(2),xrng(1)],[yrng(1),yrng(1),yrng(2),yrng(2)]);
%for i=1:izh  % braizo vejo vektorius
 %   vv=vvminh+(vvmaxh-vvminh)*zh(i)/H;
  %  quiver3(0,0,zh(i),vv(1),vv(2),vv(3),20);
%end
%t=0;r=[h0;v0];      % pradines reiksmes
```

```

ttt=[0:dt:tmax];
ix=0;  % kai pasieks zeme, taps ix=1
[TT,VV]=ode45(@fnk,ttt,v0); % PDL sprendimas
hndl=[]; % pradine markerio valdiklio reiksme
ttp=0; % laikas vizualizacijos valdymui
%vaizdavimas
%quiver3(0,0,H,v0(1),v0(2),v0(3),20,'LineWidth',3)
for i=1:length(TT)
    v=VV(i);
    h0 = h0 - dt*v;
    vaizdavimas(i, v); %vaizdavimas kiekviename zingsnyje
    if (i*dt-iskleidimas) < 0.005, disp('Isskleistas parasiutas, aukstis = '), disp(h0);end
    if h0 < 0, break,end
    if ttp==0,disp('***** Press Key'),pause,ttp=0.01;else, pause(ttp);end
    %str=sprintf('t=%5.3g x=%5.3g y=%5.3g z=%5.3g',TT(i),r(1:3));title(['\fontname{Courier}',str]);
end

return   % pagrindines programos pabaiga

% &&&&&&&&&&&&&&&&&&&&&&&&   vidines funkcijos &&&&&&&&&&&&&&&&&&&&&&&&&&&&
function dy=fnk(t,r) % Lygciu sistemos funkcija
    v=r;
    if (t > Tp) && (iskleidimas == 0), iskleidimas = t; end
    if t < Tp, Fr=-c1*v^2;
    else,    Fr=-c2*v^2;
    end
         dy(1)=Fr/m+g;
    return
end


function vaizdavimas(i, v)  % Vaizdavimo funkcija
    x=i*dt;y=v;
    if ~isempty(hndl), set(hndl,'Color',[1 1 1]*0.7);end
    if x > Tp, hndl=plot(x,y,[color,'s'],'MarkerSize',4);
    else hndl=plot(x,y,[color,'*'],'MarkerSize',8);
    end
    return
end
% &&&&&&&&&&&&&&&&&&&&&&&&   vidiniu funkciju pabaiga &&&&&&&&&&&&&&&&&&&&&

end
```

2. Išvados

Darant šį darbą buvo įsisavinti PDL sudarymo ir sprendimo metodai. Buvo prisimintas fizikos kursas.
Išnagrinėti Eulerio ir 4 eilės Rungės ir Kutos metodai, nustatytas pastarojo pranašumas tikslumo atžvilgiu prieš Eulerio metodą. To buvo galima tikėtis, nes skaičiuojant sprendinį 4 eilės Rungės ir Kutos metodu yra daromi papildomi tarpiniai skaičiavimai (prognozės, atgalinis Eulerio metodas, Simpsono koreguojanti formulė). Taip pat panagrinėta MATLAB standartinė funkcija ode45 diferencialinių lygčių sprendimui.