Paprastųjų diferencialinių lygčių sprendimas

# Užduotis

𝑚 masės sviedinys iššaunamas vertikaliai į viršų pradiniu greičiu 𝑣0 iš aukščio ℎ0. Žinoma, kad oro pasipriešinimas proporcingas sviedinio greičio kvadratui, o proporcingumo koeficientas lygus 𝑘1, kai sviedinys kyla, ir 𝑘2, kai sviedinys leidžiasi. Kokį maksimalų aukštį ir kada pasieks sviedinys? Kada sviedinys nusileis ant žemės?

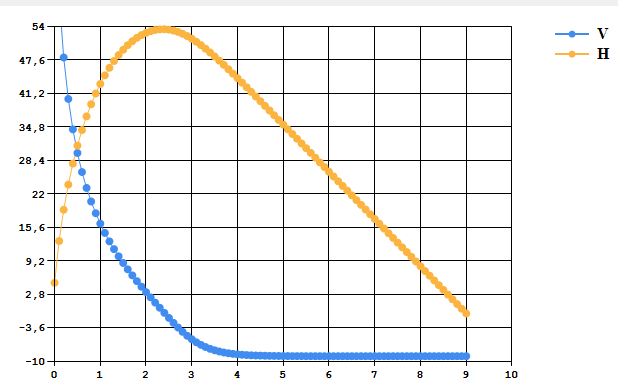
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| m, kg | v0, m/s | h0, m | k1, kg/m | k2, kg/m |
| 5 | 80 | 5 | 0,15 | 0,6 |

# Uždavinio sprendimas

Uždavinio sprendimui naudojami Niutono dinamikos dėsniai. Iš šių dėsnių galima išsireikšti tokias formules judančio kuno matematiniui modeliui apskaičiuoti:

Iš šių formulių išreiškiamos šios lygtys:

Toliau apskaičiuojami rezultatai ir nubraižomi grafikai.

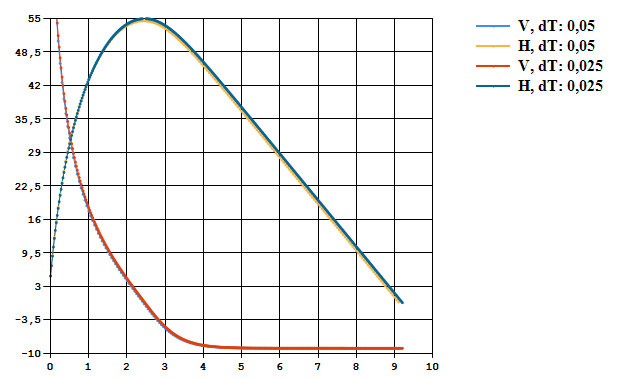


Gauti rezultatai: kai Δt = 0.1, maksimalus pasiektas aukštis yra 53,4891177 m, pasiektas po 2.3 sekundžių, o žemę pasiekė po 9 sekundžių. Šie rezultatai atitinka ir atvaizduotą grafiką.

# Sprendimo rezultatų tikslumo ir metodo stabilumo tyrimas

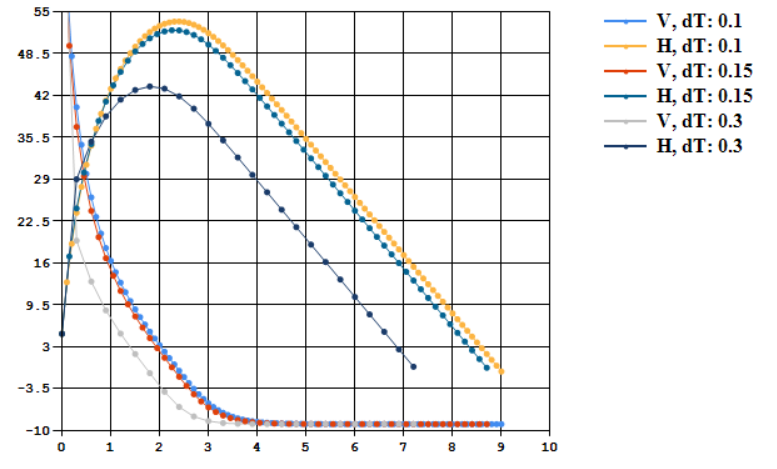
## Tikslumo tyrimas

Tikslumo tyrimas buvo atliktas palyginant grafikus sprendžiant uždavinį su skirtingais Δt. Buvo pasirinkta Δt = 0.05 ir Δt = 0.025. Kaip matome grafikai yra labai arti vienas kito, todėl galime teigti, kad jau šiame žingsnyje uždavinio sprendimo rezultatai yra pakankamai tikslūs.



## Stabilumo tyrimas

Uždavinio sprendimo stabilumo tyrimo grafikai:



Iš grafikų matome, kad uždavinio sprendinio rezultatai yra tinkami, kai metodo žingsnis Δt yra mažesnis už 0.3. Kai žingsnis didesnis, grafiko kreivė pernelyg greit tolsta nuo tikslių sprendimo rezultatų.

# Programos kodas

public void Run(double deltaT, bool keepForm = false)

{

var m = 5d;

var v = 80d;

var s = 5d;

var k1 = 0.15d;

var k2 = 0.6d;

var g = 9.8d;

var t = 0d;

form.PreparareForm(0, 10, -10, 55);

var H = form.X1X2;

var V = form.Fx;

V = form.chart1.Series.Add($"V, dT: {deltaT}");

V.MarkerStyle = MarkerStyle.Circle;

V.MarkerSize = 5;

V.ChartType = SeriesChartType.Point;

V.ChartType = SeriesChartType.Line;

H = form.chart1.Series.Add($"H, dT: {deltaT}");

H.MarkerStyle = MarkerStyle.Circle;

H.MarkerSize = 5;

H.ChartType = SeriesChartType.Point;

H.ChartType = SeriesChartType.Line;

if (!keepForm)

{

V.Points.Clear();

H.Points.Clear();

}

V.Points.AddXY(t, v);

H.Points.AddXY(t, s);

double F(double \_v)

{

var k = \_v > 0 ? k1 : k2;

return (m \* - g - k \* \_v \* Math.Abs(\_v)) / m;

}

double G(double \_v) => \_v;

while (s > 0)

{

t += deltaT;

s = s + deltaT \* G(v);

v = v + deltaT \* F(v);

H.Points.AddXY(t, s);

V.Points.AddXY(t, v);

form.richTextBox1.AppendText($"t:{t}, v:{v}, s:{s} \n");

}

}

public void RunAnalysis()

{

var t1 = 0.1d;

var t2 = 0.15d;

var t3 = 0.3d;

Run(t1);

Run(t2, true);

Run(t3, true);

}