Acta et Commentationes, Sciences of Education, nr. 2(36), 2024 ISSN 1857-0623
p. 43-51 E-ISSN 2587-3636

CZU: 517+004.6 DOI: 10.36120/2587-3636.v36i2.43-51

MODELAREA MATEMATICĂ ȘI SIMULAREA ASISTATĂ DE CALCULATOR A SISTEMELOR MECANICE CU UN GRAD DE LIBERTATE

Nicolae BALMUȘ, dr., conf. univ.
https://orcid.org/0000-0002-0491-2918
Catedra Informatică și Tehnologii Informaționale
Universitatea Pedagogică de Stat „Ion Creangă" din Chișinău

Rezumat. Modelarea matematică și simularea asistată de calculator a diferitor fenomene și procese din natură sunt activități stipulate în curriculumul preuniversitar și programele analitice universitare pentru disciplinele fizica, matematica, informatica. În acest context, în lucrare este descris și implementat în limbajul de programate Pascal un algoritm original de rezolvare numerică a ecuațiilor diferențiale de ordinul doi. Aplicațiile interactive de simulare sunt realizate în mediul de programare Delphi. În baza exemplelor, descrise detaliat în lucrare, utilizatori pot realiza activități interactive de simulare pentru orice proces, care din punct de vedere matematic se descrie printr-o ecuație diferențială de gradul doi.
Cuvinte cheie: algoritmi de calcul numeric, modelare matematică, programare vizuală, simulare asistată de calculator, fenomene oscilatorii.

MATHEMATICAL MODELING AND COMPUTER-AIDED SIMULATION OF ONE-DEGREE-OF-FREEDOM MECHANICAL SYSTEM

Abstract. Mathematical modeling and computer-assisted simulation of various phenomena and processes in nature are activities stipulated in the pre-university curriculum and university analytical programs for the disciplines of physics, mathematics, computer science. In this context, the paper describes and implements in the Pascal programming language an original algorithm for the numerical resolving of second-order differential equations. Interactive simulation applications are built in the Delphi programming environment. Based on the examples, detailed described in the paper, users can perform interactive simulation activities for any process, which from a mathematical point of view is described by a differential equation of the second degree.
Keywords: numerical computing algorithms, mathematical modeling, visual programming, computer-aided simulation, oscillatory phenomena.

1. Introducere

Din punct de vedere matematic sistemele mecanice cu un singur grad de libertate pot fi descrise printr-o ecuație diferențială ordinară de gradul doi, care în caz general are forma:

$$\ddot{q} = f(t, q, \dot{q}), \quad (1)$$

unde prin $q, \dot{q}, \ddot{q}$,– sunt notate coordonata, viteza și accelerația generalizată; t este timpul;

$$\dot{q} = \frac{dq}{dt};\ \ddot{q} = \frac{d^2q}{dt^2}.$$

Pentru simularea asistată de calculator a comportamentului acestor sisteme mecanice în funcție de timp sunt necesare metode analitice sau numerice de rezolvare a ecuației diferențiale (1), în baza condițiilor inițiale care, de regulă sunt formulate prin intermediul

coordonatelor și vitezelor generalizate ale sistemului mecanic în momentul inițial de timp (problema Cauchy).

Pentru un punct material cu masa m care se deplasează rectiliniu sub acțiunea unei forțe F, ecuația diferențială a mișcării rectilinii reiese din legea a doua a lui Newton: $ma = F(t, x, v)$ care, în baza definițiilor prin derivate după timp a vitezei $v = \dot{x}$ și a accelerației $a = \ddot{x}$ se transformă într-o ecuație diferențială ordinară de gradul doi:

$$\ddot{x} = F(t, x, \dot{x})/m \tag{2}$$

Dacă mișcarea este uniform accelerată $(a = const)$, soluțiile ecuației (2) sunt cunoscute:

$$x = x_0 + v_0 t + at^2/2;\ v = v_0 + a * t \tag{3},$$

unde x_0 $şi$ v_0 sunt condițiile inițiale (poziția și viteza inițială a punctului material).

In marea majoritate a sistemelor mecanice, studierea cărora prezintă interes științific sau metodico-didactic, ecuațiile diferențiale nu pot fi rezolvate în mod analitic. În aceste cazuri, unica modalitate de determinare a soluțiilor în funcție de timp este rezolvarea numerică a ecuațiilor diferențiale de tipul (1).

Scopul principal al acestei lucrări constă în identificarea unui algoritm de rezolvare numerică a ecuațiilor diferențiale de tipul (1) efectiv, suficient de exact și stabil pentru un interval de timp comparabil cu durata fenomenelor reale care se studiază (minute, ore).

2. Metode directe pentru rezolvarea numerică a ecuațiilor diferențiale ordinare de ordinul doi

Metode directe pentru rezolvarea numerică a ecuațiilor diferențiale ordinare de ordinul doi nu există [1],. De regulă aceste ecuații se transformă în sisteme de ecuații de ordinul unu. Dacă notăm $\dot{x} = v$ ecuația (3) se transformă într-un sistem din două ecuații diferențiale de ordinul unu:

$$\dot{x} = v(t, x);\ a) \quad \dot{v} = f(t, x, v);\ b); \tag{5}$$

pentru care există diverse metode numerice de determinare a soluțiilor *x* și *v* la pasul t_{i+1} exprimate prin condițiile inițiale, considerate cunoscute la pasul t_i . O încercare de utilizare a acestei metode pentru rezolvarea problemelor de dinamică cu multe grade de libertate a fost realizată în lucrarea [2], dar din cauză că funcția $v(t, x)$ nu este cunoscută în formă explicită, precizia calculelor nu a fost înaltă.

Prezentăm în continuare un algoritm de rezolvarea numerică directă a unei ecuații diferențiale de ordinul doi de tipul (3) în care soluțiile pentru variabila v se determină în baza metodei Runge Kutta de ordinul IV iar soluțiile pentru variabila *x*, în baza dezvoltării în seria Tylor, în care sau neglijat terminii $0(h^3)$.

$$v_{i+1} = v_i + (K_0 + 2K_1 + 2K_2 + K_3)/6 \tag{6}$$

$$K_0 = hf(t_i, x_i, v_i)$$

$$K_1 = hf\left(t_i + \frac{h}{2}, x\left(t_i + \frac{h}{2}\right), v_i + K_0/2\right)$$

$$K_2 = hf\left(t_i + \frac{h}{2}, x\left(t_i + \frac{h}{2}\right), v_i + K_1/2\right)$$
$$K_3 = hf(t_i + h, x(t_i + h), v_i + K_2)$$
$$x(t_i + h) = x_i + v_i h + \frac{f(t_i, x_i, v_i)h^2}{2} \quad (7)$$

În baza acestui algoritm vom implementa o procedură Pascal-Delphi care, în baza condițiilor inițiale (*t0,x0,v0*) va determina soluțiile (*x, v*) ale ecuației diferențiale (3a) în momentul de timp *t* cu o precizie *ε* impusă apriori. Calculele în procedură se realizează în două etape:

Etapa1 - în baza formulelor (6-7) se determină soluțiile *xc* și *vc* pentru pasul *h=t-t0.* Aceste soluții sunt considerate soluții precedente pentru următoarea etapă care este una iterativă;

Etapa2 este un proces iterativ care începe cu transferul soluțiilor obținute la etapa1 în variabilele xp și vp și continuă cu un ciclu repetitiv în care intervalul de integrare *h=t-t0* se divizează succesiv în 2,4,8, etc subintervale. Pe fiecare subinterval soluțiile se calculează conform formulelor (7-8) în care condițiile inițiale sunt considerate soluțiile obținute la pasul precedent. Procesul iterativ se consideră finalizat la pasul n pentru care diferențele în valoare absolută dintre soluțiile curente și precedente devin mai mici sau egale cu precizia impusă *eps*. Prezentăm în continuare codul complet al acestei proceduri implementată în mediul de programare Delphi:

```
procedure RED2RK4vP(eps,t0,x0,v0:extended;pf:pfxv;t:extended;
var x,v:extended);
//type  pfxv=procedure (t,x,v:extended; var f:extended); - tip
procedural declarat
//global, în baza căruia se calculează funcția
//eps – precizia; t0,x0,v0 – condițiile inițiale; x,v – soluțiile în
momentul de timp t;
var n,i:integer;f,h,xc,xp,vc,vp,xx0,vv0,tt0,k0,k1,k2,k3:extended;
var xh2,xh:extended;
begin //Etapa1
    h:=t-t0;  n:=1;
    pf(t0,x0,v0,f);  k0:=h*f;  xh2:=x0+v0*h/2+f*sqr(h/2)/2;
    pf(t0+h/2,xh2,v0+k0/2,f);k1:=h*f;
    pf(t0+h/2,xh2,v0+k1/2,f);k2:=h*f;
    xh:=x0+v0*h+f*sqr(h)/2;   pf(t0+h,xh,v0+k2,f);k3:=h*f;
    vc:=v0+(k0+2*k1+2*k2+k3)/6;   xc:=x0+v0*h+(vc-v0)*h/2;
    repeat //Etapa2
        xp:=xc;vp:=vc;  h:=h/2;n:=n*2;
        xx0:=x0;vv0:=v0;tt0:=t0;
        for i:=1 to n do begin
           pf(tt0,xx0,vv0,f);k0:=h*f;
            xh2:=xx0+vv0*h/2+f*sqr(h/2)/2;
```

```
            pf(tt0+h/2,xh2,vv0+k0/2,f);k1:=h*f;
            pf(tt0+h/2,xh2,vv0+k1/2,f);k2:=h*f;
            xh:=xx0+vv0*h+f*sqr(h)/2;
            pf(tt0+h,xh,vv0+k2,f);k3:=h*f;
            vc:=vv0+(k0+2*k1+2*k2+k3)/6;
            xc:=xx0+vv0*h+(vc-vv0)*h/2;
            xx0:=xc; vv0:=vc;tt0:=tt0+h;
        end;
    until (abs(xc-xp)<eps)and (abs(vc-vp)<eps);
    x:=xc; v:=vc;
end;
```

3. Validarea, prin experiment numeric, a algoritmului de rezolvare numerică a ecuațiilor diferențiale de ordinul doi

Sistemul mecanic format dintr-un punct material de masă m suspendat printr-o legătură rigidă de lungime l în câmpul omogen de gravitație al pământului (numit pendul gravitațional ideal) este unul din cele mai bine studiate sisteme mecanice (din punct de vedere experimental și teoretic). Vom utiliza acest sistem pentru validarea algoritmului și testarea procedurilor de rezolvare numerică a ecuațiilor diferențiale ordinare de ordinul doi.

Ecuația diferențială care descrie oscilațiile pendulului gravitațional simplu poate fi dedusă prin diverse metode. În cazul când în calitate de coordonată generalizată este ales unghiul φ de abatere a pendulului din poziția de echilibru, ecuația are forma [3]:

$$\ddot{\varphi} = -\frac{g}{l}\sin(\varphi) \qquad (8)$$

În caz general ecuația (8) nu are soluții analitice, exprimate prin funcții elementare, care ar putea fi utilizate pentru simularea oscilațiilor pendulului gravitațional. Unica soluție pentru simularea în timp real a acestui proces oscilatoriu constă în rezolvarea numerică a ecuației (8) cu precizie înaltă, relativ stabilă în timp.

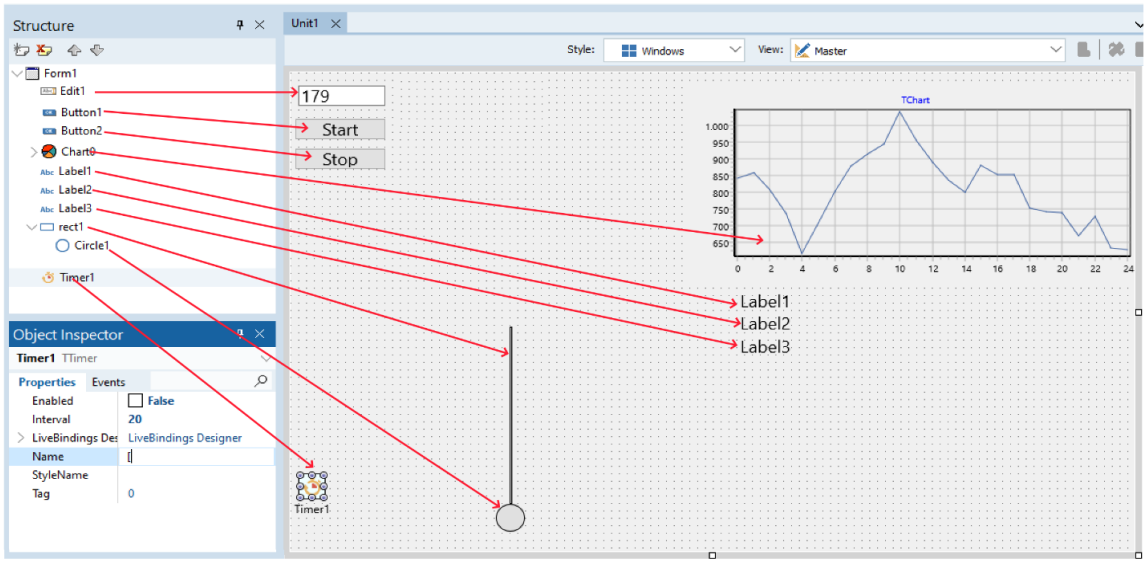

Figura 1. Designul aplicației Delphi
pentru simularea oscilațiilor pendulului gravitațional

Interfața aplicației Delphi în care vom realiza simularea în timp real a oscilațiilor pendulului gravitațional este reprezentată în figura 1. În obiectul *Edit1* se setează unghiul inițial de abatere a pendulului de la poziția inițială de echilibru, în grade. Firul pendulul se modelează cu ajutorul componentei *Rect1* iar bila prin componenta *Circle1*, care la nivel de design se setează ca copil al componentei *Rect1*. Centrul de rotire a componentei *Rect1* este setat la nivel de design prin valorile *RotationCenter.X=0.5*, *RotationCenter.Y=0*. Componenta *Chart0* se utilizează pentru vizualizarea graficului oscilațiilor. În componentele *Label1*, 2, 3 se afișează, respectiv: timpul curent al animației, eroarea relativă a simulării și intervalul real de timp pe parcursul căruia se rezolvă ecuația diferențială. Pentru componenta *Timer1* proprietatea *Enabled* se setează la valoarea *False* iar *Interval* la valoarea 20 milisecunde.

Procesul de simulare a oscilațiilor începe în momentul când utilizatorul apasă *Button1* (*Start*) în care se execută următorul cod:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
    t0:=0;v0:=0; g:=10;l:=1;
    // Lungimea pendulului se setează la valoare 1m, g=10 m²/s.
    x0:=edit1.Text.ToExtended*pi/180;//valoarea inițială a unghiului de
    abatere a pendulului în radiani este preluat di componenta Edit1.
    ti:=now;// timpul inițial se setează la valoarea ceasului de sistem.
    Et0:=g*(l-l*cos(x0));
    //Se calculează energia mecanică  în momentul inițial de timp
    timer1.Enabled:=true;
    //Activează procedura Timer1Timer, descrisă în continuare.
    chart0.Series[0].Clear; //
end;
```

În evenimentul OnTimer al timer-lui Timer1 se realizează următorul cod:

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
    t:=(now-ti)*24*3600;//Calculează momentul de timp pentru care
    urmează să fie
    //rezolvată numeric ecuația diferențială pentru condițiile inițiale
    preluate din Button1.
    RED2RK4vP(1E-12,t0,x0,v0,pfn,t,x,v); //Se apelează procedura care
    rezolvă ecuația
    //diferențială descrisă în procedura pfn pentru momentul de timp t.
    //Rezultatele sunt în variabilele x și v, valorile cărora se
    utilizează pentru realizarea
    //rotiri pendulului și calcularea erorii sistemului la momentul
    curent.
    label3.Text:= (t-t0).ToString;
    rect1.RotationAngle:=x*180/pi;
```

```
    Et:=g*(l-l*cos(x))+sqr(v)/2;//  Se  calculează  energia  mecanică
    curentă.
    t0:=t;x0:=x; v0:=v; // Se modifică condițiile inițiale
    label1.text:='t(h:m:s)='+timetostr(t/24/3600);
    label2.text:='(E-E0)/E0*100%='+((et/et0-1)*100).ToString;
    chart0.series[0].AddXY(t,x);
end;
```

Forma explicită a ecuației diferențiale (8) se implementează în următoarea procedură:

```
procedure pfn(t,x,v:extended; var f:extended);
begin
    f:=-(g/l)*sin(x); //expresia (8) transcrisă
end;
```

Notă: *Instrucțiunile* ***t0:=t; x0:=x; v0:=v;*** *sunt foarte importante. Ele modifică condițiile inițiale pentru următorul apel al procedurii care rezolvă numeric ecuația diferențială. Momentul de timp curent și soluțiile calculate la etapa curentă sunt considerate condiții inițiale pentru următoarea etapă. Dacă nu se realizează aceste instrucțiuni procedura care rezolvă ecuația diferențială necesită din ce în ce mai mult timp pentru asigurarea preciziei impuse, eps. Ca urmare, animația foarte repede încetează să funcționeze.*

În figura 2 este reprezentată secvența de simulare a oscilațiilor pendulului gravitațional în momentul de timp 57minute și 45 secunde.

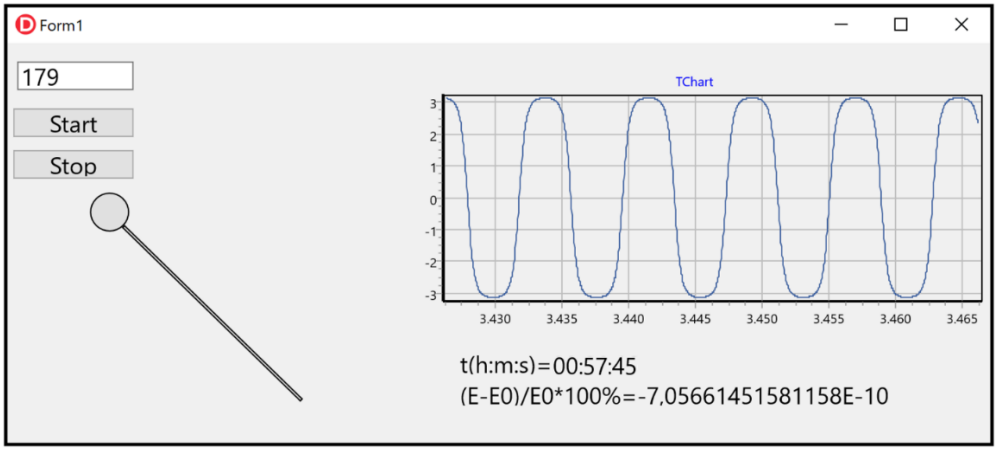

Figura 2. Simularea oscilațiilor pendulului gravitațional (captură de ecran)

Deoarece ecuația diferențială (8) nu are soluții analitice, exprimate prin funcții elementare, corectitudinea rezolvării numerice se verifică indirect, prin intermediul verificării legii conservării energiei mecanice:

$$E = E_p + E_c = mgl(1 - \cos(\varphi)) + mv^2/2 = const \qquad (9)$$

Energia mecanică inițială E_0 se calculează în procedura Button1Click, în baza condițiilor inițiale. În procedura Timer1Timer, pentru fiecare moment de timp al animației, în baza soluțiilor numerice pentru φ și v se calculează $E(t)$. În fereastra aplicației se

afișează eroarea relativă, în procente $\varepsilon = \left(\frac{E(t)}{E_0} - 1\right) 100\%$ (figura 2). Pe parcursul intervalului de timp de aproximativ o oră această mărime nu a depășit valoarea 10^{-10}.

Acest rezultat confirmă, indirect corectitudinea calculelor și eficiența algoritmului de rezolvare numerică directă a ecuațiilor diferențiale de ordinul doi, descris în paragraful precedent.

4. Modelarea matematică și simularea asistată de calculator a unor sisteme mecanice cu un grad de libertate

4.1. *Pendulul gravitațional real*

În calitate de pendul gravitațional real vom considera sistemul mecanic compus dintr-un corp de formă sferică de rază *R* și masă *m* suspendat printr-o legătură rigidă ideală (masa căreia se neglijează) de lungime *l* asupra căruia acționează forța de gravitație $\vec{P} = m\vec{g}$ și forța de rezistență a mediului ambiant $\vec{R}$. În calitate de model pentru forța de rezistență care acționează asupra corpului de formă sferică vom utiliza forța Stokes $\vec{R} = -6\pi\, R\vec{v}$, unde η este coeficientul de vâscozitate dinamică.

În baza acestor aproximații, ecuația diferențială care descrie sistemul mecanic primește forma:

$$I\ddot{\varphi} = mglsin(\varphi) - 6\pi\eta l R^2\dot{\varphi}, \tag{11}$$

unde I este momentul de inerție al sistemului mecanic, calculat, față de punctul de suspensie:

$$I = m(l+R)^2 + \frac{2}{5}\pi m R^2 \tag{12}$$

Aplicația Delphi cu ajutorul căreia se simulează oscilațiile acestui sistem mecanic este analogică cu cea descrisă în figura 1, cu următoarele completări:

1. În evenimentul Button1Click se setează condițiile inițiale $(\varphi_0, \dot{\varphi}_0)$ și valorile parametrilor sistemului *m, L, R*, η (niu) fie direct, fie prin intermediul casetelor de dialog de tip Edit; În baza acestor valori se calculează momentul de inerție *I* conform formulei (12): J:=m*sqr(L+R)+(2/5)*pi*m*sqr(R).
2. În baza ecuației diferențiale (11) se perfectează procedura *pnf*:

```
procedure pfn(t,x,v:extended;var f:extended);
begin
    f:=(-m*g*L*sin(x)-6*pi*niu*L*sqr(R)*v)/J;// se perfectează în baza
    expresiei (11); J este momentul de inerţie care se calculează în
    procedura Button1Click.
end;
```

În figura 3 este reprezentată o captură de ecran a procesului de simulare.

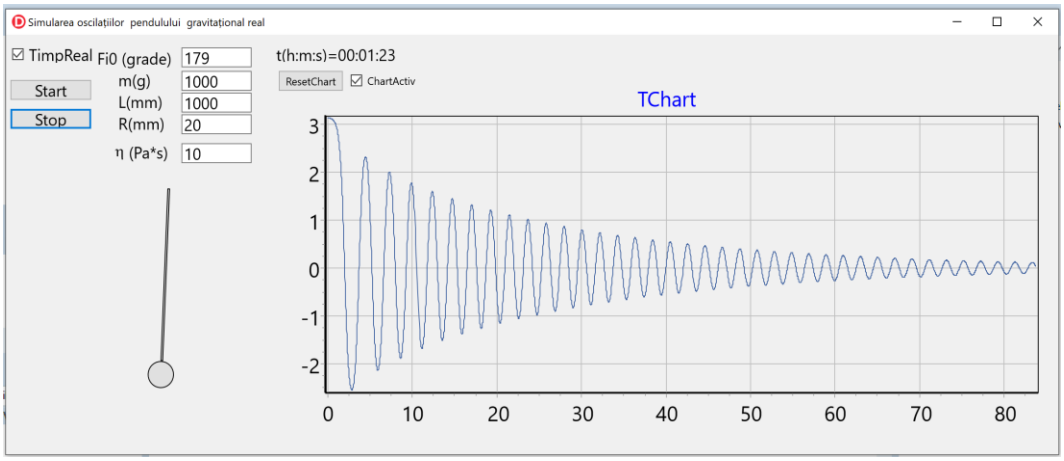

Figura 3. Simularea oscilațiilor amortizate ale unui pendul gravitațional real

4.2. *Oscilații parametrice*

În calitate de sistem mecanic în care se produc oscilații parametrice vom considera un pendul gravitațional, punctul de suspensie al căruia efectuează oscilații armonice în direcție verticale. Din punct de vedere matematic acest sistem se descrie prin următoarea ecuație diferențială de ordinul doi [4]:

$$\ddot{\varphi} = -\left(\frac{g}{l} - \frac{h}{l}\omega^2 \sin(\omega t)\right)\sin(\varphi), \qquad (13)$$

unde h și ω sunt amplitudinea și frecvența ciclică a oscilațiilor armonice ale punctului de suspensie a pendulului.

Aplicația Delphi cu ajutorul căreia se simulează oscilațiile acestui sistem mecanic este analogică cu cea descrisă în figura 1, cu următoarele completări:

1. În evenimentul Button1Click se setează condițiile inițiale $(\varphi_0, \dot{\varphi}_0)$ și valorile parametrilor sistemului h și ω fie direct, fie prin intermediul casetelor de dialog de tip Edit,
2. În baza ecuației diferențiale (11) se perfectează procedura *pnf*:

```
procedure pfn(t,x,v:extended;var f:extended);
begin
    f:=-(g/l-h/l*sqr(w)*sin(w*t))*sin(x);
end;
```

În figura 4 sunt indicați parametrii sistemului oscilant și capturile de ecran în care se observă:

A. Fenomenul de rezonanță parametrică;
B. Oscilațiile pendulului în jurul poziției superioare de echilibru;
C. Oscilațiile pendulului în jurul poziției inferioare de echilibru;

Modificând parametrii sistemului oscilant, utilizatorul prin experiment numeric descoperă:

- condițiile pentru care se observă rezonanța parametrică (w=2w_0) și cum influențează amplitudinea excitatorului h asupra acestui proces;

- care sunt condițiile pentru care pendulul efectuează oscilații în poziție inversată $\frac{h\,\omega}{l\omega_0} > \sqrt{2}$, demonstrată teoretic în [5].

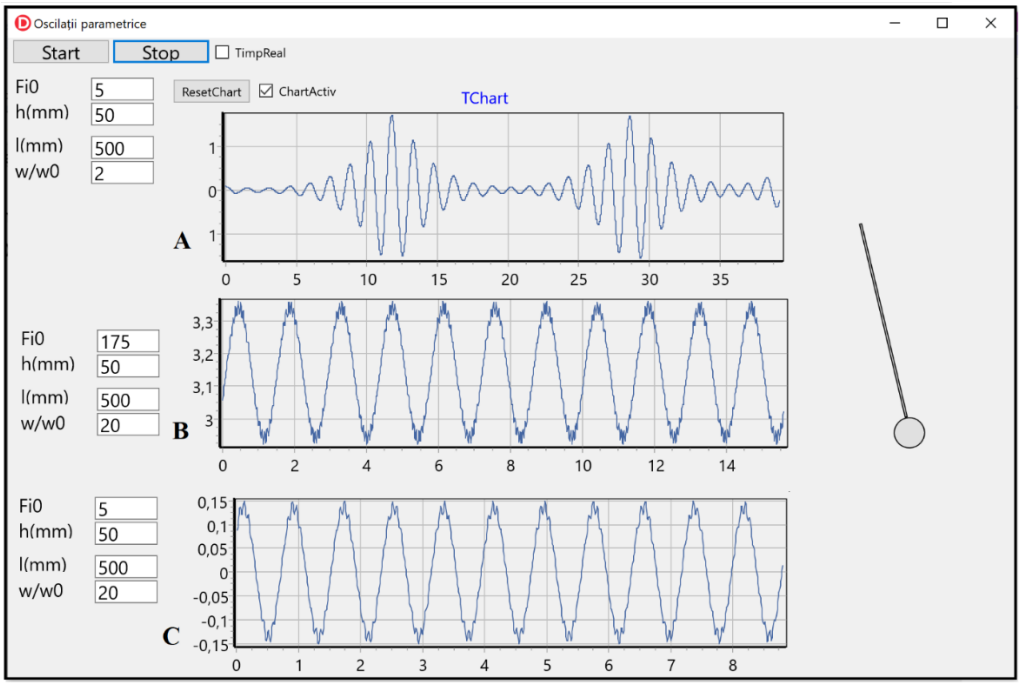

Figura 4. Simularea oscilațiilor parametrice ale pendulului gravitațional

Concluzii

În baza investigațiilor realizate și a codurilor de programare Pascal, prezente în lucrare, utilizatorul final (elevii, studenții, masteranzii, cadrele didactice), cu eforturi minimale și cunoștințe medii de programare Delphi poate restabili aplicațiile descrise și dezvolta în continuare alte aplicații de modelare-simulare de concepție proprie. Aceste activități vor contribui substanțial la înțelegerea mai profundă a fenomenelor și proceselor fizice studiate și vor dezvolta competențe de creare a software-lor educaționale cu ajutorul cărora utilizatorii vor realiza experimente și lucrări virtuale de laborator.

Bibliografie

1. KENDALL, E. A., WEIMIN, H., DAVID, S. *Numerical Solution of Ordinary Differential Equations*. John Wiley & Sons, Inc, 2009.
2. BALMUȘ, N., BALMUȘ, I. Modelarea matematică şi elaborarea software-lor pentru simularea în timp real pe calculator a problemelor de dinamică cu multe grade de libertate. În: *Meridian ingineresc*. 2009, nr.1, p. 59-67.
3. NECULAI, A. *Eseu despre Fundamentele Modelarii Matematice*. București: Editura Academiei Romane, 2012.
4. STRUBLE, R. Oscillations of a pendulum under parametric excitation. În: *Quart. Appl. Math*. nr. 21 (1963), pp. 121-131.
5. КАПИЦА, П. Л. Динамическая устойчивость маятника при колеблющейся точке подвеса. În: *ЖЭТФ*. 1951, nr. 21, pp. 588 – 597.