

# Fizică experimentală

## Laborator 9

Victor E. Ambruş

*Facultatea de fizică, Universitatea de Vest din Timişoara,  
Bd. Vasile Pârvan nr. 4, Timişoara, RO 300223, România*

15 decembrie 2017

### Rezumat

Pe parcursul acestui laborator, ne vom familiariza cu comenzile pentru rezolvarea ecuaţiilor algebrice şi a celor diferenţiale, atât prin metode analitice, cât şi numeric. Vom considera mişcarea circulară şi traiectoria proiectilului în mediu rezistiv şi nerezistiv.

## 1 Mişcarea circulară

### 1.1 Definirea unei funcţii

Legea de mişcare a mişcării circulare este:

$$\mathbf{r}(t) = x(t)\mathbf{i} + y(t)\mathbf{j}, \quad (1)$$

unde  $x(t)$  şi  $y(t)$  sunt funcţii armonice de  $t$ :

$$x(t) = R \cos \omega t, \quad y(t) = R \sin \omega t. \quad (2)$$

Putem defini ecuaţiile de mişcare de mai sus sub forma unor funcţii:

```
xp(t) := R * cos(t * omega);  
yp(t) := R * sin(t * omega);
```

Mai sus am definit funcţiile  $xp(t)$  şi  $yp(t)$ , care descriu dependenţa de timp a coordonatelor  $x$  şi  $y$  ale particulei. Pentru definirea unei funcţii se foloseşte operatorul `:=` (de remarcat că *variabilele* se definesc folosind operatorul `:`).

### 1.2 Instrucţiunea *subst*

Putem afişa traiectoria corespunzătoare ec. (2) folosind următoarea comandă:

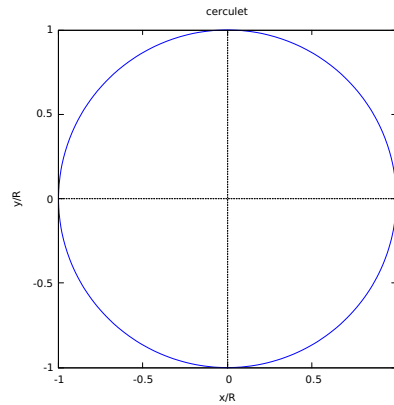


Figura 1: Traiectoria circulară corespunzătoare ec. (2).

```
plot2d([parametric,
        subst([R=1, omega=2*pi], xp(t)),
        subst([R=1, omega=2*pi], yp(t)),
        [t,0,1]],
        [xlabel,"x/R"],[ylabel,"y/R"],
        [title, "Traiectorie circulara"],
        [same_xy, true]);
```

Să analizăm comanda de mai sus.

- Comanda `plot2d` generează un grafic folosind `gnuplot`.
- Primul argument este `{[parametric, x, y, [t, tmin, tmax]]}`. Graficele parametrice reprezintă curba având coordonatele  $(x, y)$  pentru valorile parametrului  $t$  specificat pe poziția a patra, care ia valori între `tmin` și `tmax`.
- Coordonatele  $x$  și  $y$  sunt date de funcțiile `xp(t)` și `yp(t)`, însă aceste funcții au fost definite de parametri nespecificați `R` și `omega`. Pentru a putea reprezenta grafic traiectoria aferentă acestor funcții, trebuie să dăm valori numerice lui  $R$  și  $\omega$ .
- Putem atribui valori numerice acestor parametri folosind comanda `subst`. Comanda `subst` ia doi parametri: primul parametru reprezintă o listă de atribuire de valori de tipul `[var1 = val1, var2 = val2, ...]`, în timp ce parametrul al doilea reprezintă expresia în care se fac substituțiile.
- Funcția `plot2d` mai acceptă parametri care pot fi folosiți pentru configurarea descrierii axelor graficului (`[xlabel, "x/R"]` și `[ylabel, "y/R"]`) sau a titlului graficului (`[title, "Traiectorie circulara"]`).

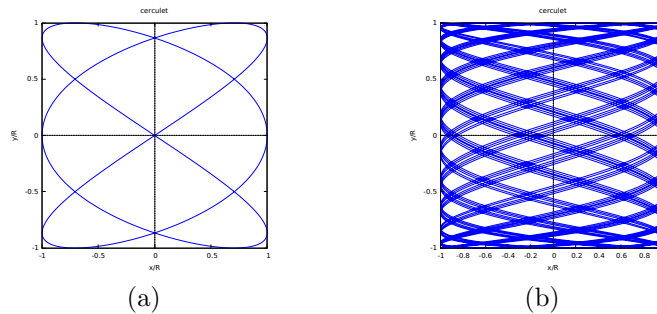


Figura 2: Figurile Lissajous care rezultă din utilizarea unor pulsații diferite pe axele  $x$  și  $y$  în ec. (2). (a)  $\omega_x = 6\pi$ ,  $\omega_y = 4\pi$ ; (b)  $\omega_x = 2\pi^2$ ,  $\omega_y = 2\pi$ .

- Ultimul parametru al comenzii `plot2d` este `[same_xy, true]`, care impune aceeași scală pe cele două axe (astfel încât traiectoria se va vedea într-adevăr ca un cerc).

### 1.3 Instrucțiunea *block*

Să considerăm acum o mișcare plană care reprezintă compunerea a două oscilații cu perioade și amplitudini nu neapărat egale. Mai exact, ne interesează următoarea traiectorie:

$$x(t) = R_x \cos(\omega_x t), \quad y(t) = R_y \cos(\omega_y t). \quad (3)$$

Traectoria rezultantă poartă numele de *figură Lissajous*. Putem reprezenta această traiectorie folosind funcțiile `xp(t)` și `yp(t)`, însă dând valori diferite pentru `R` și `omega` cu ajutorul comenzii `subst`.

Să presupunem că dorim să încercăm combinații multiple ale valorilor lui  $R_x$ ,  $R_y$ ,  $\omega_x$  și  $\omega_y$ . În acest caz, este convenabilă utilizarea unei structuri de tip `block`, în care să putem defini valorile acestea folosind niște variabile locale:

```
block([Rx,Ry,omegax,omegay],
      Rx:1,Ry:1,omegax:6*pi,omegay:4*pi,
      plot2d([parametric,
               subst([R=Rx,omega=omegax],xp(t)),
               subst([R=Ry,omega=omegay],yp(t)),
               [t,0,1]],
              [xlabel,"x/R"],[ylabel,"y/R"],
              [title, "Compunere"],
              [nticks,1000],
              [same_xy, true])
);
```

Mai sus am definit un bloc de instrucțiuni folosind comanda `block`. Pe prima poziție am specificat lista de *variabile locale*, și anume `Rx`, `Ry`, `omegax` și `omegay`.

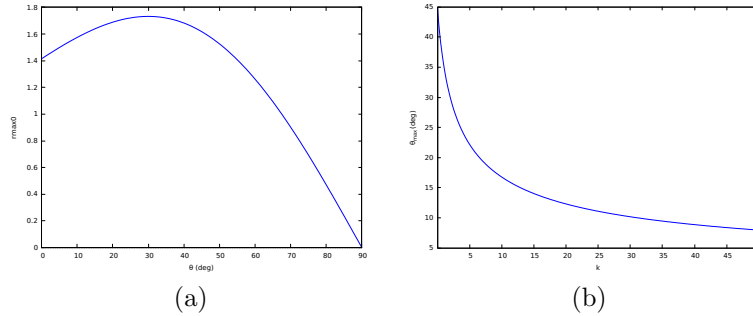


Figura 3: (a) Valoarea maximă a bătaii adimensionalizate  $r_{\max 0}$  ca funcție de  $\theta$  pentru  $k = 2$ ; (b) Valoarea unghiului  $\theta$  sub care este lansat proiectilul pentru a obține bătaia maximă ca funcție de  $k$ .

Mai departe am utilizat comanda `plot2d` discutată anterior, de data aceasta substituind  $R = R_x$  și  $\omega = \omega_x$  pentru coordonata orizontală  $x_p(t)$ , respectiv  $R = R_y$  și  $\omega = \omega_y$  pentru coordonata verticală  $y_p(t)$ . Am mai specificat prin parametrul `[nticks, 1000]` să fie folosite 1000 de puncte pentru generarea traiectoriei, pentru a evita aspectul colțuros al graficului.

**Problema 1.** Să se utilizeze instrucțiunile de mai sus pentru a reprezenta figura Lissajous corespunzătoare compunerii a două mișcări armonice de amplitudini  $R_x = R_y = 1$ , având pulsațiile  $\omega_x = 2\pi^2$  și  $\omega_y = 2\pi$ . Să se folosească domeniul temporal  $t \in [0, 20]$ . [Vezi fig. 2(b).]

## 2 Aruncarea oblică

Pentru studierea mișcării oblice, definim legea de mișcare a unui corp sub acțiunea unei accelerații constante în sensul negativ al axei  $y$ :

```
xp(t) := x0 + v0 * cos(theta) * t;
yp(t) := y0 + v0 * sin(theta) * t - g * t^2 / 2;
```

**Problema 2.** Să se găsească momentul de timp  $t$  la care proiectilul atinge solul ( $y = 0$ ).

Soluție:

```
solve(yp(t)=0,t);
```

Se observă că instrucțiunea de mai sus returnează două valori (una pozitivă și una negativă). Pentru a salva coordonata acestui punct, putem folosi instrucțiunea:

```
tmax:rhs(%[2]);
```

Comanda `rhs` returnează membrul drept al unei egalități, în timp ce `%[2]` se referă la elementul al doilea din șirul returnat în urma execuției ultimei comenzi.

Cele două comenzi de mai sus (`solve` și `rhs`) pot fi combinate într-o singură comandă care returnează direct valoarea lui `tmax`:

```
tmax: rhs(solve(yp(t)=0,t)[2]);
```

**Problema 3.** Să se găsească distanța față de punctul de aruncare când proiectilul atinge solul (bătaia).

Soluție: Distanța parcursă până la atingerea solului (bătaia) se poate calcula folosind `tmax`:

```
rmax: xp(tmax)-x0;
```

**Problema 4.** Să se găsească momentul de timp la care înălțimea proiectilul este maximă. Să se atribuie această valoare variabilei `tymax`. [R:

```
tymax = (sin(theta)*v0)/g]
```

**Problema 5.** Să se găsească înălțimea maximă pe care o atinge proiectilul. Să se atribuie această valoare variabilei `ymax`. [R:

```
ymax = y0+(sin(theta)^2*v0^2)/(2*g)]
```

În continuare, este convenabil să introducem o scală  $h_0$  pentru înălțime, definită ca fiind înălțimea la care energia potențială a proiectilului este egală cu energia sa cinetică inițială:

$$h_0 = \frac{v_0^2}{2g}. \quad (4)$$

Cu ajutorul acestei scale de lungime, putem defini  $y_0 = kh_0$  ( $k$  fiind înălțimea inițială în unități de  $h_0$ ). Cu ajutorul lui  $h_0$  și  $k$ , putem defini bătaia adimensională ca raportul  $r_{\max}/2h_0$ , după cum urmează:

```
rmax0: radcan(subst([v0=sqrt(2*g*h0),y0=k*h0],rmax)/(2*h0));
```

Funcția `radcan` a fost introdusă pentru a simplifica expresia rezultantă.

**Problema 6.** Să se reprezinte grafic `rmax0` ca funcție de unghiul  $\theta$ , exprimat în grade, sub care proiectilul este aruncat pentru cazul când  $k = 2$ . [Vezi fig. 3(a).]

Soluție:

```
plot2d(subst([k=2,theta=thdeg*pi/180],rmax0),[thdeg,0,90],
[xlabel,"{/Symbol q} (deg)"],
[ylabel,"r_{max, 0}"],
[title,"Bataia adimensionala"]);
```

Comanda de mai sus face în instrucțiunea `subst` o conversie a lui  $\theta$  din valoarea exprimată în grade (`thdeg`) într-o valoare exprimată în radiani ( $\theta \times \pi/180^\circ$ ), iar rezultatul se atribuie necunoscutului `theta`. Descrierea axei  $x$  este  $\theta(\text{deg})$ , unde simbolul grecesc  $\theta$  se obține în `gnuplot` folosind instrucțiunea `{/Symbol q}`.

**Problema 7.** Să se găsească valoarea lui  $\theta$  pentru care `rmax0` este maxim pentru cazul când  $k = 2$ . Să se exprime rezultatul în grade.

Soluție: În acest caz, comanda `solve` nu va găsi analitic valoarea lui  $\theta$  pentru care  $r_{\max,0}$  este maxim. De aceea, vom folosi comanda `find_root`:

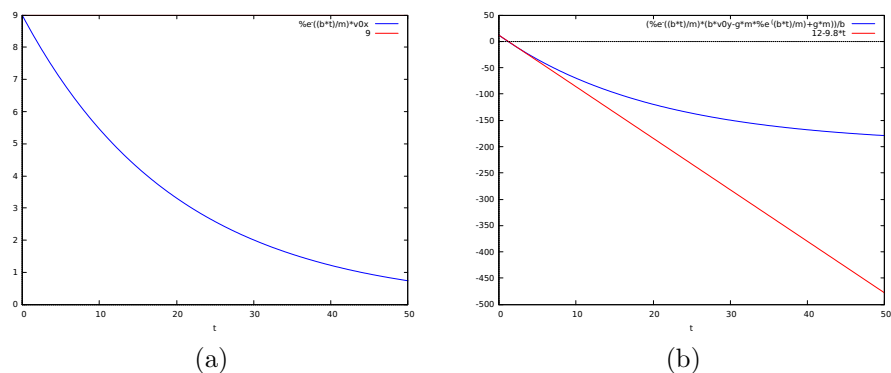


Figura 4: Comparație între componentele orizontale (a) și verticale (b) ale vitezelor în aruncarea oblică cu rezistență liniară și fără rezistență.

```
float(find_root(diff(subst(k=2,rmax0),theta),theta,0,%pi/2)*180/%pi);
=> 30.0
```

Putem reprezenta grafic valoarea lui  $\theta$  (în grade) la care  $r_{\max 0}$  este maxim ca funcție de  $k$ :

```
plot2d(float(find_root(diff(rmax0,theta),theta,0,%pi/2)*180/%pi),
[k,0.01,50],[xlabel,"k"],[ylabel,"{/Symbol q}_{max}(deg)"]);
```

Rezultatul acestei comenzi se poate vizualiza în fig. 3(b).

### 3 Aruncarea oblică în mediu cu rezistivitate liniară

Să considerăm problema aruncării oblice în cazul când proiectilul întâmpină o forță de rezistență proporțională cu viteza:

$$m \frac{d\mathbf{v}}{dt} = -mg\mathbf{j} - b\mathbf{v}, \quad (5)$$

unde  $b$  e o constantă care depinde de forma obiectului și de proprietățile mediului. Ecuația are soluție analitică:

$$\mathbf{v} = \mathbf{v}_0 e^{-bt/m} - \frac{mg}{b} \left(1 - e^{-bt/m}\right) \mathbf{j}. \quad (6)$$

În continuare vom folosi **Maxima** pentru a obține această soluție.

Pentru început, să definim variabilele  $dv_x$  și  $dv_y$  care exprimă ecuațiile diferențiale pe axele  $x$  și  $y$  aferente ec. (5):

```
dvx:m*'diff(vx(t),t)=-b*vx(t)$
dvy:m*'diff(vy(t),t)=-m*g-b*vy(t)$
```

Atenție! Uneori apostroful care precede comanda `diff` nu este copiat cu comanda copy. Este vorba despre caracterul `'`, care se găsește pe tastatură lângă tasta Enter.

Să observăm că dependența funcțiilor `vx` și `vy` de variabila `t` este indicată explicit. Pentru a evita evaluarea prematură a operatorului diferențial, acesta este precedat de o ghilime dreaptă. Suplimentăm ecuațiile de mai sus introducând condițiile inițiale  $\mathbf{v}(t=0) = \mathbf{v}_0$ :

```
atvalue(vx(t),t=0,vx0)$
atvalue(vy(t),t=0,vy0)$
```

Comanda care rezolvă ecuațiile diferențiale `dvx` și `dvy` este `desolve`:

```
radcan(desolve([dvx,dvy],[vx(t),vy(t)]));
```

Pe prima poziție apar ecuațiile într-o listă, iar pe a doua poziție apar funcțiile care trebuie determinate.

**Problema 8.** Pentru cazul aruncării oblice în mediu cu rezistivitate liniară, să se determine:

1. Valoarea asimptotică a vitezei când  $t \rightarrow \infty$ , folosind comanda `limit`.
2.  $\mathbf{x}(t)$ , rezolvând ecuația diferențială  $\frac{d\mathbf{x}}{dt} = \mathbf{v}$  împreună cu condițiile inițiale  $\mathbf{x}(t=0) = 0$ .

Pentru a compara evoluția vitezei în timp față de cazul fără rezistență, mai întâi definim variabilele care țin vitezele din cazul curent:

```
vlinx:rhs(%o??[1])$
vliny:rhs(%o??[2])$
```

Semnele de întrebare se referă la numărul de ordine al rezultatului rulării comenzii `desolve`.

```
block([m:1,b:0.05,vx0:9],
      plot2d([vlinx,vx0],[t,0,100])
      );

block([m:1,b:0.05,vy0:12,g:9.8],
      plot2d([vliny,vy0-g*t],[t,0,50])
      );
```

Rezultatele comenzilor de mai sus pot fi vizualizate în fig. 4

## 4 Aruncarea oblică în mediu cu rezistivitate pătratică

Să considerăm că forța de rezistență depinde de pătratul vitezei particulei:

$$m \frac{d\mathbf{v}}{dt} = -mg\mathbf{j} - c\mathbf{v}\mathbf{v}. \quad (7)$$

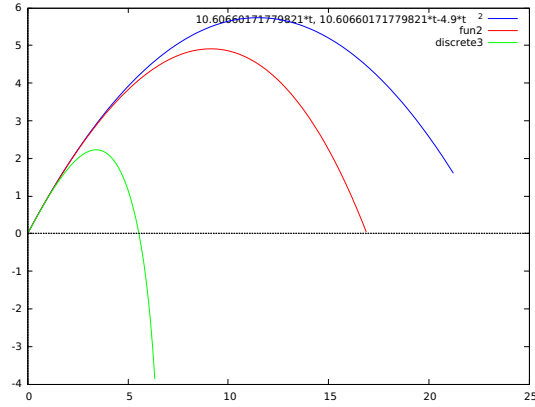


Figura 5: Graficul traiectoriei ( $y$  în funcție de  $x$ ) pentru aruncarea oblică în cazul fără rezistență, cu rezistență liniară ( $b = 10^{-6}$ ) și cu rezistență pătratică ( $c = 10^{-6}$ ). Viteza inițială are modulul  $v_0 = 15$ , unghiul de proiectare este de  $45^\circ$  iar masa este  $4,2 \times 10^{-6}$ .

Deoarece nu se cunoaște soluția analitică a acestei ecuații, suntem nevoiți să recurgem la metode numerice. În primul rând, trebuie definite mărimile care intervin în problemă:

```
v0:15$ deg:%pi/180.0$ theta:45.0*deg$
m:4.2e-6$ c:1e-6$ g:9.8$
v0x:float(v0*cos(theta))$ v0y:float(v0*sin(theta))$
```

Modulul vitezei la  $t = 0$  este  $v_0 = 15$ , unghiul de aruncare este de  $45^\circ$ , masa particulei este de  $4,2 \times 10^{-6}$  iar coeficientul de rezistență este  $c = 10^{-6}$ . Variabila `deg` are valoarea  $\pi/180^\circ$ , fiind utilă pentru conversia între grade și radiani. Soluția numerică se poate obține folosind algoritmul Runge-Kutta de ordinul 4, pe care îl apelăm cu funcția `rk`:

```
tmax:2$
data:rk([vx,vy,-c*sqrt(vx^2+vy^2)*vx/m,-g-c*sqrt(vx^2+vy^2)*vy/m],
        [x,y,vx,vy],[0,0,v0x,v0y],[t,0,tmax,tmax/100])$
vals:makelist([data[i][2],data[i][3]],i,1,length(data))$
```

În comanda `rk`, pe prima poziție este specificată o listă de ecuații care vor fi rezolvate numeric simultan. Pe poziția a doua apare lista de variabile ( $x$ ,  $y$ ,  $v_x$  și  $v_y$ ), iar pe poziția a treia apare lista cu valorile inițiale pentru aceste variabile. Pe ultima poziție se specifică variabila de integrare ( $t$ ) împreună cu domeniul acesteia (între 0 și  $t_{\max}$ ), respectiv pasul temporal sau distanța între două valori succesive ale variabilei de integrare (în cazul nostru, folosim 100 de puncte, deci pasul de timp este  $t_{\max}/100$ ). Pentru compararea traiectoriei cu rezultatele de mai înainte, putem rula comenzile:



```

xliber:v0x*t$
yliber:v0y*t-g*t^2/2$

b:1e-6$
xlin:(%e^(-(b*t)/m)*(m*%e^((b*t)/m)-m)*v0x)/b$
ylin:(%e^(-(b*t)/m)*(-g*m^2+(g*m^2-b*g*m*t)*%e^((b*t)/m)+
(b*m*%e^((b*t)/m)-b*m)*v0y))/b^2$
plot2d([[parametric,xliber,yliber,[t,0,tmax]],
[parametric,xlin,ylin,[t,0,tmax]],
[discrete,vals]]);

```

Rezultatul poate fi vizualizat în Fig. 5.