5.9. Calcul diferențial

Conceptul fundamental de *diferențială* reprezintă un instrument foarte puternic în modelarea fenomenelor în știință și inginerie. El se bazează pe analiza creșterii sau descreșterii funcțiilor în raport cu argumentele lor. Diferențiala unei funcții $y = f(x)$ notată cu dy, $df(x)$ se numește produsul dintre derivata ei și variația (elementară a) argumentului:

$$dy = f'(x) \cdot dx,$$

unde dx este diferențiala argumentului funcției.

Calculul diferențial se bazează pe trei teoreme fundamentale: teorema lui Roll, teorema lui Lagrange și teorema lui Cauchy și pe conceptele de monotonie (creștere și descreștere) a funcțiilor și de extremum (minim și maxim).

În cazul funcțiilor de mai multe variabile de forma $z = f(x, y, ...)$ se definește *diferențiala totală* $dz = f'_x(x, y, ...)dx + f'_y(x, y, ...)dy + ...$, în care $d_x z = f'_x dx$, $d_y z = f'_y dy$ și așa mai departe sunt *diferențialele parțiale*.

5.9.1 Rezolvarea ecuațiilor diferențiale

Se numește *ecuație diferențială ordinară*[1] o ecuație care conține argumentul, funcția căutată a acestui argument și derivatele ei de diferite ordine, de forma generală: $F(x, y, y', y'', ..., y^{(n)}) = 0$. Ordinul cel mai mare al derivatei funcției căutate, din ecuația dată, se numește *ordinul* ecuației diferențiale. Orice funcție care, fiind înlocuită împreună cu derivatele sale de ordine corespunzătoare, într-o ecuație diferențială în locul funcției căutate și a derivatelor sale, transformă această ecuație în identitate (satisface identic ecuația) se numește *soluția* (*integrala*) ecuației diferențiale. Procesul de rezolvare a ecuațiilor diferențiale se numește *integrare*.

[1] Ecuațiile diferențiale ordinare reprezintă clasa cea mai generală de ecuații diferențiale alături de cea a ecuațiilor diferențiale cu derivate parțiale și de cea a ecuațiilor integrale.

Scopul integrării unei ecuaţii diferenţiale este obţinerea unei funcţii care satisface ecuaţia dată şi îndeplineşte anumite *condiţii iniţiale* impuse. În urma integrării ecuaţiilor diferenţiale se obţin *soluţii particulare* (în baza teoremei de existenţă a lui Cauchy). Ansamblul tuturor soluţiilor particulare se numeşte *soluţie generală*. Graficul soluţiei ecuaţiei diferenţiale se numeşte *curbă integrală* a acestei ecuaţii. Soluţia generală a unei ecuaţii diferenţiale este reprezentată de o *familie de curbe* integrale.

În multe situaţii, soluţia ecuaţiilor diferenţiale poate fi obţinută pe cale analitică, adică prin integrare exactă cu ajutorul primitivelor funcţiei de integrat şi a condiţiilor iniţiale date. De cele mai multe ori însă, în practică se apelează la metode de *integrare numerică* din următoarele motive:

- funcţia *f* nu admite primitivă sau un poate fi descompusă în forme care se integrează exact;
- intervenţia omului în rezolvarea analitică a problemei nu este posibilă;
- integrarea numerică este o tehnică generală destinată automatizării calculului pentru rezolvarea oricărei ecuaţii diferenţiale.

În practica inginerească, multe fenomene fizice pot fi descrise cu ajutorul *ecuaţiilor diferenţiale de ordinul întâi* de forma:

$$y' = \frac{dy}{dx} = f(x, y), \tag{5.9.1}$$

unde x este considerată *variabila independentă* (de exemplu timpul), iar y este *funcţia necunoscută*. Funcţia f este o expresie liniară sau neliniară, poate să conţină explicit variabila independentă x, sau nu. Aceasta este funcţia care se integrează în procesul de rezolvare a ecuaţiei diferenţiale.

Condiţia iniţială se exprimă generic în forma:

$$y(x_0) = y_0 \tag{5.9.2}$$

şi are un rol important în rezolvarea ecuaţiei diferenţiale (determinarea soluţiei particulare). Soluţia generală a unei ecuaţii diferenţiale de ordinul întâi (relaţia 5.9.1) conţine o constantă arbitrară C, care constituie *parametrul* familiei de curbe integrale şi se determină în vederea obţinerii de soluţii particulare cu ajutorul condiţiei iniţiale (relaţia 5.9.2). Prin urmare, condiţia iniţială capătă o importanţă deosebită în problemele inginereşti, practic fiind determinată de starea iniţială a sistemelor sau de anumite condiţii la limită impuse.

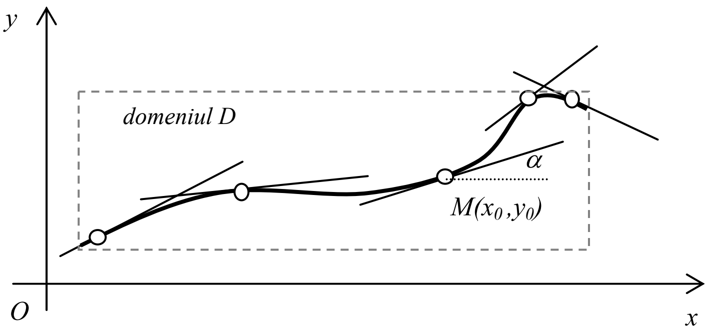
Observație

Soluția unei ecuații care nu conține constanta arbitrară, deci ale cărei soluții particulare nu se obțin din soluția generală se numeşte soluție *singulară*[2].

Interpretare grafo-analitică

Interpretarea grafo-analitică a rezolvării unei ecuații diferențiale de forma (5.9.1) se face pe baza graficului funcției *f* definită într-un domeniu D în planul xOy. Se defineşte un câmp de direcții pe D, astfel încât în fiecare punct $M(x,y) \in D$ se ia direcția dată de tangenta $\mathrm{tg}\,\alpha = \dfrac{dy}{dx} = f(x,y)$, unde unghiul $\alpha = \mathrm{arctg}\ f(x,y)$ este unghiul format de direcția tangentei cu sensul pozitiv al axei Ox.

În figura de mai jos se ilustrează interpretarea grafo-analitică pentru soluția particulară dată de condiția inițială (5.9.2) $M(x_0, y_0)$.



A. Funcții pentru integrarea numerică a ecuațiilor diferențiale

Rezolvarea numerică a ecuațiilor diferențiale se bazează pe conceptul de calcul diferențial, în care variațiile (diferențialele) variabilelor trec din domeniul infiniților mici în domeniul finit sub forma diferențelor finite obținute prin discretizarea domeniului numeric continuu cu o anumită rezoluție dată prin *pasul de integrare*. Din acest motiv rezolvarea numerică a ecuațiilor diferențiale utilizează metode aproximative în procesul de estimare a valorilor funcției pe intervalul de integrare dat. În tabelul următor

[2] În toate punctele curbei care reprezintă soluția singulară, condițiile teoremei de existență a lui Cauchy nu sunt satisfăcute.

sunt prezentate în sinteză categoriile de metode utilizate la rezolvarea numerică a ecuaţiilor diferenţiale.

Clasa	Metoda
Metode numerice directe ("cu paşi separaţi")	Metoda lui Euler (şi variante) Metoda lui Taylor Metode Runge-Kutta Metoda Moulton
Metode numerice indirecte ("cu paşi legaţi")	Metoda Adams-Bashforth Metoda predictor-corector
Metode analitice cu soluţii aproximative	Metoda lui Picard Metoda seriilor de puteri Metoda seriilor lui Taylor Metoda inegalităţilor diferenţiale

Esenţa integrării numerice

Vom explica mecanismul rezolvării numerice a ecuaţiilor diferenţiale pe baza celei mai simple (şi intuitive) metode: *metoda lui Euler*, aparţinând metodelor numerice directe.

Ne propunem să integrăm ecuaţia diferenţială (5.9.1) cu condiţia iniţială (5.9.2) pe intervalul $[x_0,b]$. Algoritmul este următorul:

- Se discretizează domeniul variabilei independente, considerând noduri echidistante cu pasul *h*: $x_i = x_{i-1} + h$, rezulând *n* puncte de evaluare a funcţiei $x_0 < x_1 < \ldots < x_i < \ldots < x_n = b$.
- Se estimează valoarea aproximativă a soluţiei în nodul x_1 pe baza ecuaţiei $\frac{\Delta y}{\Delta x} \cong f(x,y)$, care reprezintă practic aproximarea cu diferenţe finite a ecuaţiei diferenţiale (1), deci:

$$- \; y_1 = y_0 + f(x_0,y_0)(x_1 - x_0)\text{, adică } y_1 = y_0 + f_0 h.$$

- Se repetă recursiv procedeul până la epuizarea valorilor indicelui *i*:

$$- \; y_i = y_{i-1} + f(x_{i-1}, y_{i-1})h.$$

- În final se obţin valorile estimate ale soluţiei particulare a ecuaţiei diferenţiale pe intervalul dat: $y_0, y_1, \ldots, y_n$.

Metoda lui Euler se mai numeşte şi *metoda liniilor poligonale*. Din punct de vedere practic are avantajul că este simplă şi rapidă dar si dezavantajul unei slabe precizii, inclusiv pe seama acumulării erorilor de la paşii anteriori. Din acest motiv, în aplicații se folosesc metode de tip Euler îmbunătățite: metoda Euler modificată, metoda Euler-Cauchy, metoda Euler-Heun, metoda Euler-Fox.

Metoda Euler este o *metodă numerică directă* deoarece *la fiecare pas, valoarea curentă a funcției* y_i *se calculează printr-o relație de recurență numai în funcție de valoarea precedentă* y_{i-1}.

În fapt, se constată că metoda lui Euler reprezintă un caz particular al metodei lui Taylor, care este tot o metodă numerică directă ce foloseşte aproximarea funcțiilor prin descompunerea lor în seria cu acelaşi nume. Astfel, cu notațiile folosite până acum, soluția ecuației diferențiale de forma (5.9.1) se poate exprima sub forma seriei Taylor:

$$y(x_i) = y(x_{i-1}) + h\, y'(x_{i-1}) + \frac{h^2}{2!} y''(x_{i-1}) + \dots + \frac{h^p}{p!} y^{(p)}(x_{i-1}) + \dots$$

Ordinul metodei de integrare este dat de ordinul maxim al derivatei care se reține în dezvoltarea Taylor. Astfel, pentru $p = 1$ se obține chiar metoda Euler, care este deci o metodă de ordinul întâi:

$$y(x_i) = y(x_{i-1}) + h\, y'(x_{i-1})$$

Metodele Runge-Kutta de diferite ordine reprezintă de asemenea cazuri particulare ale metodei lui Taylor ce folosesc dezvoltări de diferite ranguri, care dau şi ordinul metodei $p = 2,3,4,5,\dots$.

Fără a intra în detalii vom face mențiunea că metodele Runge-Kutta nu necesită evaluarea nici unei derivate în procesul de calcul, utilizând doar valoarea soluției găsită la pasul anterior. Acest lucru se face prin estimarea variației curente a soluției sub forma unei combinații liniare de valori cunoscute de la pasul anterior de forma $\Delta y_{i-1} = \sum_{j=1}^{p} c_j k_j$, unde $c_j \in R$, iar coeficienții k_j se calculează cu anumite formule particulare în funcție de ordinul metodei. Cu alte cuvinte, diferențiala funcției în punctul curent de calcul este asimilată cu corecția finită care se calculează cu ajutorul valorilor funcției f în anumite puncte din intervalul (x_{i-1}, x_i).

Formulele concrete pentru calculul corecției pentru diferite ordine ale metodei au fost deduse astfel încât valoarea estimată a soluției $y_i = y_{i-1} + \Delta y_{i-1}$ să coincidă cu dezvoltarea ei în serie Taylor până la termenii de rang egal cu ordinul metodei.

De exemplu, pentru metoda Runge-Kutta de ordinul 3 estimarea corecției se face cu formula:

$$\Delta y_{i-1} = \frac{h}{6}\left(k_1 + 4k_2 + k_3\right),$$

în care:

$$k_1 = f(x_{i-1}, y_{i-1})$$

$$k_2 = f\left(x_{i-1} + \frac{1}{2}h,\ y_{i-1} + \frac{1}{2}hk_1\right)$$

$$k_3 = f\left(x_{i-1} + h,\ y_{i-1} - k_1 + 2k_2\right)$$

Funcțiile implementate în MATLAB pentru integrarea numerică a ecuațiilor diferențiale operează direct cu ecuații de ordinul întâi (la care pot fi reduse şi cele de ordin superior) şi folosesc algoritmi îmbunătățiți ai metodei de bază Runge-Kutta. Programele realizate pentru aceste metode sunt destul de complexe şi permit modificarea internă a pasului de integrare *h*, adaptarea ordinului metodei, efectuarea unor operații interne de interpolare şi extrapolare.

În comparație cu metodele de tip Euler, metodele de tip Runge-Kutta sunt mai laborioase, necesitând un timp de calcul cu atât mai mare cu cât ordinul metodei este mai ridicat. Acest dezavantaj este compensat totuşi de faptul că precizia metodei este deosebit de bună, fapt ce permite adoptarea unui pas de integrare mai mare şi obținerea unui timp de calcul rezonabil.

Principala eroare de metodă care se manifestă în cazul metodelor Runge-Kutta se numeşte *eroare de trunchiere*, datorată în principal trunchierii seriei Taylor.

Ansamblul funcțiilor Matlab pentru integrarea numerică a ecuațiilor diferențiale prin metode directe sunt prezentate în tabelul 5.6.

Tab. 5.6 Funcții Matlab pentru ecuații diferențiale

Funcția	Descriere
`ode23`	Rezolvă ecuații diferențiale *mai puțin complicate* printr-o metodă Runge-Kutta de *ordin redus* (2 sau 3). Include mecanisme de interpolare polinomială de ordinul 3 şi de extrapolare locală.
`ode23t`	Rezolvă ecuații diferențiale de *dificultate medie* precum şi *ecuații diferențiale algebrice* folosind *regula trapezelor*. Include mecanisme de interpolare.
`ode23s`	Rezolvă ecuații diferențiale de *dificultate mare* printr-o metodă Runge-Kutta de *ordin redus* (2 sau 3) implementată printr-un algoritm modificat mai nou, propus de Rosenbrock. Include mecanisme de interpolare. Extraploarea locală nu se efectuează.
`ode23tb`	Rezolvă ecuații diferențiale de *dificultate mare* printr-o metodă Runge-Kutta de *ordin redus* (2 sau 3). Metoda lucrează în doi paşi: prima dată se aplică regula trapezoidală, apoi se aplică metoda Runge-Kutta de ordin 2 printr-un mecanism denumit *backward differentiation formula (BDF)*. Metoda este propusă de Bank şi Rose şi oferă un estimator de eroare îmbunătățit şi o evaluare mai bună a soluției. Include mecanisme de interpolare.
`ode15s`	Rezolvă ecuații diferențiale de *dificultate mare* precum şi *ecuații cu diferențe finite* folosind o metodă cu *pas cvasiconstant* bazată pe mecanismul *backward differentiation formula (BDF)* cu formule de diferențiere de ordine 1, 2, ..., 5. Include mecanisme de interpolare. Extrapolarea locală nu se efectuează.
`ode45`	Rezolvă ecuații diferențiale *mai puțin complicate* printr-o metodă Runge-Kutta de *ordin mediu* (4 sau 5). Include mecanisme de interpolare polinomială de ordinul 4 şi de extrapolare locală.

Dintre *metodele numerice indirecte* pentru integrarea ecuațiilor diferențiale utilizate în Matlab menționăm metoda Adams-Bashforth, care se regăseşte implementată în funcția `ode113` sub forma algoritmului Adams-Bashforth-Moulton, cunoscut şi sub denumirea de *predictor-corector*.

Menționăm că în cazul *metodelor numerice indirecte* (cu *paşi legați* sau *multipas*) *la fiecare pas, valoarea curentă a funcției* y_i *se calculează printr-o relație de recurență în funcție de valorile precedente calculate în paşii anteriori* $y_0, y_1, ..., y_{i-1}$.

În principiu, metoda predictor-corector exploatează mai rațional valorile deja calculate ale soluției oferind precizie superioară şi timp de calcul mai bun față de metodele directe de tip Runge-Kutta. Metoda funcționează astfel:

- se face o evaluare preliminară (grosieră) a soluției ecuației cu o metodă numerică directă (de regulă Runge-Kutta), obținând un tabel cu date de tipul următor:

$$\frac{x_0 \quad x_1 \ldots \quad x_n}{y_0 \quad y_1 \ldots \quad y_n}$$

- se aplică pasul *predictor* (cu metoda Adams-Bashforth) care oferă o estimație a soluției în punctul curent: y_i;
- se aplică pasul *corector* (cu metoda Adams-Moulton) care se iterează (se rafinează de k=1,2,….) de mai multe ori până la coincidența în mod satisfăcător a valorilor succesive obținute pentru soluția în punctul curent: $y_i^k \cong y_i^{k+1}$.

Matlab pune la dispoziție următoarea funcție bazată pe o metodă numerică indirectă:

Funcția	Descriere
`Ode113`	Rezolvă ecuații diferențiale *mai puțin complicate* prin metoda numerică indirectă Adams-Bashforth-Moulton (predictor-corector) utilizând familii de formule de *ordin variabil* (între 1 şi 12). Include mecanisme de interpolare polinomială de ordinul 3 şi de extrapolare locală.

Utilizarea funcțiilor Matlab

Funcțiile *ode* (abreviere de la *Ordinary Diferential Equation*) se pot apela cu una din sintaxele:

```
        [x,y]  = ode23(odefun,xspan,y0)
        [x,y]  = ode23(odefun,xspan,y0,options)
        [x,y]  = ode23(odefun,xspan,y0,options,p1,p2,...)
[x,y,xe,ye,ie]  = ode23(odefun,xspan,y0,options...)
            sol  = ode23(odefun,[x0 xfinal],y0...)
```

unde variabilele de intrare sunt:

`odefun` – variabilă şir ce implementează funcția de integrat $f(x,y) = y'$ ca structură de date odefun(x,y) ce returnează un vector coloană cu valorile primei derivate a funcției de integrat y'. Structura de date `odefun` se implementează ca fişier Matlab de tip `function`.

`xspan` – este o structură de date de tip vector de forma xspan=[`x0 xfinal`] ce precizează intervalul de integrare între limitele variabilei independente: `x0` - valoarea inițiala, respectiv `xfinal` - valoarea finală a variabilei x.

`y0` – un vector coloană conținând condițiile inițiale.

Notă. Dacă se dorește obținerea soluției în anumite puncte, structura de date `xspan` se exprimă astfel `xspan=[x0,x1,…,xfinal]`, precizând explicit punctele de interes.

`options` – o structură de date ce permite accesul la opțiuni de integrare ce sunt definite într-un set de proprietăți implementate de funcția odeset. Aceste proprietăți sunt detaliate în tabelul 5.7.

Tab. 5.7. Setul de proprietăți implementate cu `odeset`

Proprietatea	Semnificație/Valoari	Observații
RelTol	Toleranța erorii relative Implicit: 1e-3	Se aplică tuturor componentelor vectorului soluție y(i) conferind implicit o precizie de 0.1% pentru orice metoda de integrare. Eroarea estimată la fiecare pas de integrare este: e(i)≤ max(RelTol*abs(y(i)),AbsTol(i))
AbsTol	Toleranța erorii absolute Implicit: 1e-6	Se aplică tuturor componentelor vectorului soluție y(i) pentru orice metoda de integrare.
NormControl	Controlul erorii relative în funcție de norma soluției [on | {off}]	Setată pe 'on' algoritmul controlează eroarea la fiecare pas de integrare astfel încât: norm(e) ≤max(RelTol*norm(y),AbsTol), unde norm(y) are semnificația unei distanțe euclidiene.
Refine	Factor de rafinare a rezultatului de ieșire Implicit: 1 cu excepția ode45 unde este 4.	Această proprietate crește numărul punctelor de ieșire cu valoarea specificată având ca efect netezirea rezultatelor. Nu se aplică dacă length(XSPAN) >2 sau metoda ODE returnează soluția ca pe o structură.
OutputFcn	Permite instalarea unei funcții de ieșire [function]	Funcția de ieșire precizată prin nume va fi apelată după fiecare pas de integrare. Dacă funcția de integrare ode este apelată fără argumente de ieșire OutputFcn trimite implicit la funcția odeplot. În caz contrar OutputFcn returnează implicit la lista de ieșire [ ].
OutputSel	Indici de Selecție a ieșirii [vector de intregi]	Acest vector de indici specifică care componentă a vectorului soluție este transferată Către OutputFcn. Implicit sunt transferate toate componentele
Stats	Afișează statistica efortului de calcul [on | {off}]	
Jacobian	Funcția Jacobiană [function | constant matrix]	Stabilește această proprietate la o funcție FJac (dacă FJac(t,y) returnează dF/dy) sau la o valoare constantă a lui dF/dy.
JPattern	Forma (conținutul) Jacobianului cu valori zero. [sparse matrix]	Stabilește această proprietate la o matrice cu conținut de valori zero (sparse matrix) S cu S(i,j) = 1 dacă componenta i a lui F(t,y) depinde de componenta j a lui y, și 0 în caz contrar.
Vectorized	Funcția ODE vectorizată [on | {off}]	Setează această proprietate 'on' dacă funcția ODE F este codificată așa încât F(t,[y1 y2 ...]) returnează [F(t,y1) F(t,y2) ...].

Tab. 5.7. *(Continuare)*

Events	Localizează evenimente [function]	Pentru a detecta evenimente se precizează o funcție [function] de eveniment.
Mass	Matricea de masă [constant matrix | function]	În probleme de forma M*y' = f(t,y) proprietatea se setează la *constant matrix*. Pentru probleme în care matricea de masă depinde de timp sau starea sistemului, proprietatea se setează cu o funcție care evaluează matricea de masă M.
MStateDependence	Dependența matricei de masă de y. [none | {weak} | strong]	Pentru probleme de tipul M(t)*y' = F(t,y) se setează 'none'. În cazul în care M(t,y) se folosesc setările 'weak' and 'strong'. Opțiunea 'weak' implică metode aproximative în rezolvarea ecuațiilor algebrice.
MassSingular	Matricea masă singulară [yes | no | {maybe}]	Se setează 'no' dacă matricea de masă nu este singulară.
MvPattern	Forma conținutului matricei dMv/dy cu valori zero [sparse matrix]	Setează această proprietate la o matrice S cu S(i,j) = 1 dacă pentru oricare k, componenta (i,k) a matricei M(t,y) depinde de componenta j a lui y, si 0 în caz contrar.
InitialSlope	Panta inițială a soluției yp0 [vector]	yp0 satisface M(t0,y0)*yp0 = F(t0,y0).
InitialStep	Pasul inițial sugerat [positive scalar]	Metoda de integrare va încerca cu această valoare la început. Implicit determinarea pasului inițial se face automat.
MaxStep	Limita superioară a pasului de integrare [positive scalar]	MaxStep este implicit o zecime din intervalul xspan, pentru toate metodele ode.
BDF	Utilizarea Backward Differentiation Formulas în ODE15S [on | {off}]	Această proprietate specifică dacă Backward Differentiation Formulas (metoda Gear) va fi folosită în ODE15S în locul Numerical Differentiation Formulas implicite.
MaxOrder	Ordinul maxim în cazul metodei ODE15S [1 | 2 | 3 | 4 | {5}]	

`P1, P2,…` – sunt valori de intrare ce pot fi transferate ca argumente cu rol de parametru funcției `odefun(X,Y,P1,P2...)`, precum şi tuturor funcțiilor specificate optional prin structura de date `options`.

`X, Y, XE, YE` şi `IE` –variabile de ieşire returnate de funcția de integrare numerică:

- fiecare rând în tabloul soluțiilor Y corespunde unei valori din vectorul coloană al variabilei independente X;
- XE este un vector coloană care conține valorile variabilei (argumentului) X la care s-au produs evenimente (detectate cu proprietatea Events);
- YE vector cu valorile soluției în punctele în care s-au produs evenimente;

- IE specifică evenimentele care au avut loc.

`sol` – este o structură de date, care conține evaluarea soluției în orice punct din intervalul `[X0 XFINAL]`.

Pentru a exemplifica modul de utilizare a funcțiilor `ode`, se prezintă calculul numeric al soluțiilor ecuației diferențiale de ordinul întâi $y' = 5x + 3\frac{x^2}{2}$, ce descrie o mişcare uniform accelerată cu a=3 m/s^2 şi cu viteza inițială v_0=5 m/s, pe intervalul de timp [0,4], cu condiția inițială y(0)=0. În acest exemplu variabila independentă x are semnificația de timp, iar soluția numerică (x,y) descrie legea spațiului în mişcarea uniform accelerată. Evident, în acest exemplu simplu problema admite soluția analitică $y_a = \frac{5}{2}x^2 + \frac{1}{2}x^3$.

Soluția numerică (x,y) şi cea analitică (x,y_a) se vor reprezenta comparativ pe acelaşi grafic, cu markere respectiv cu linie continuă.

<u>Paşii de rezolvare sunt următorii</u>:

(1) Se creează un fişier funcție cu numele **f.m**, care defineşte ecuația diferențială:

```
function yprim=f(x,y)
yprim=0+5*x+3*x^2/2;
```

(2) Fişierul funcție creat se apelează cu una dintre sintaxele pentru funcțiile `ode`, de exemplu:

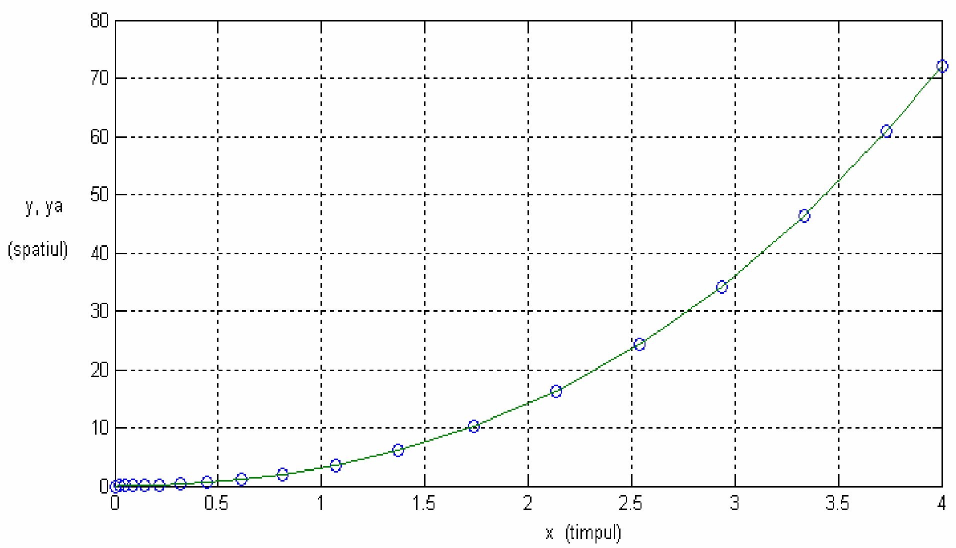
```
[x,y]=ode23(@f,[0 4],0)
```

care returnează soluția numerică prezentată în tabelul de mai jos, ce conține şi soluția analitică `ya=(5/2)*x.^2+(1/2).*x.^3`, în acest caz fiind identică cu cea numerică.

Argument funcție	Soluția numerică	Soluția analitică
x =	y =	ya =
0	0	0
0.0250	0.0016	0.0016
0.0500	0.0063	0.0063
0.0873	0.0194	0.0194
0.1414	0.0514	0.0514
0.2163	0.1221	0.1221
0.3163	0.2660	0.2660
0.4460	0.5416	0.5416
0.6103	1.0448	1.0448
0.8149	1.9305	1.9305
1.0659	3.4457	3.4457
1.3703	5.9813	5.9813
1.7363	10.1538	10.1538
2.1363	16.2838	16.2838
2.5363	24.2393	24.2393
2.9363	34.2122	34.2122
3.3363	46.3945	46.3945
3.7363	60.9782	60.9782
4.0000	72.0000	72.0000

Rezultatele se reprezintă grafic cu instrucțiunile:

```
plot(x,y,'o',x,ya);
grid
```



În continuare, vom aborda aceeaşi problemă de mai sus pentru a exemplifica integrarea funcţiei cu parametri prin utilizarea sintaxei următoare:

```
[x,y] = ode23(odefun,xspan,y0,options,p1,p2,...)
```

În ecuaţia diferenţială de forma $y' = v_0 \cdot x + a \cdot \frac{x^2}{2}$ considerăm parametrii $p_1 = v_0$ şi $p_2 = a$ care vor fi transferaţi funcţiei $f(x, p_1, p_2)$ prin valori numerice concrete. Etapele de rezolvare sunt:

- se scrie fişierul **f.m** astfel:

```
function yprim=f(x,y,p1,p2)
yprim=p1*x+p2*x^2/2;
```

- se apelează funcţia de integrare numerică cu valori concrete pentru argumentele de tip parametru, de pildă p1=0 şi p2=3:

```
[x,y]=ode23(@f,[0 4],0,[],0,3)
```

Rezultatul reprezintă în acest caz evoluţia spaţiului în mişcarea uniform accelerată cu p2=3m/s^2 şi viteză iniţială nulă (p1=0).

<u>*Notă*</u>. Pe poziţia argumentului `options` se introduce [ ] dacă setul de opţiuni rămâne cu valorile implicite.

Exemple.

(1) Să se integreze numeric ecuaţia diferenţială de ordinul întâi cu parametru $y' = -p \cdot y$, pe intervalul [0,3] cu condiţia iniţială y(0)=5.

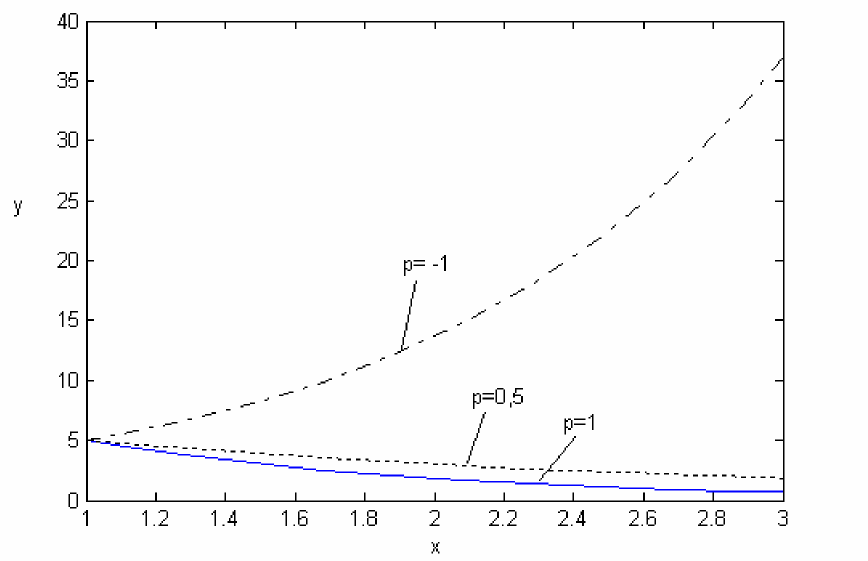
Rezolvare.
- Se creează fişierul funcţie **g.m**:

```
function yprim=g(x,y,p)
yprim=-p*y;
```

- Se apelează succesiv funcția pentru integrare numerică `ode23` pentru trei valori ale parametrului p, astfel:

```
[x,y]=ode23(@g,[1 3],5,[],1)
[x,y]=ode23(@g,[1 3],5,[],0.5)
[x,y]=ode23(@g,[1 3],5,[],-1)
```

Se reprezintă grafic soluțiile numerice pentru cele trei cazuri.



(3) Să se integreze numeric ecuația diferențială de ordinul întâi cu parametri $y' = p_1 \cdot y + e^{p_2 \cdot x}$, pe intervalul [-1,2] cu condiția inițială y(0)=1.

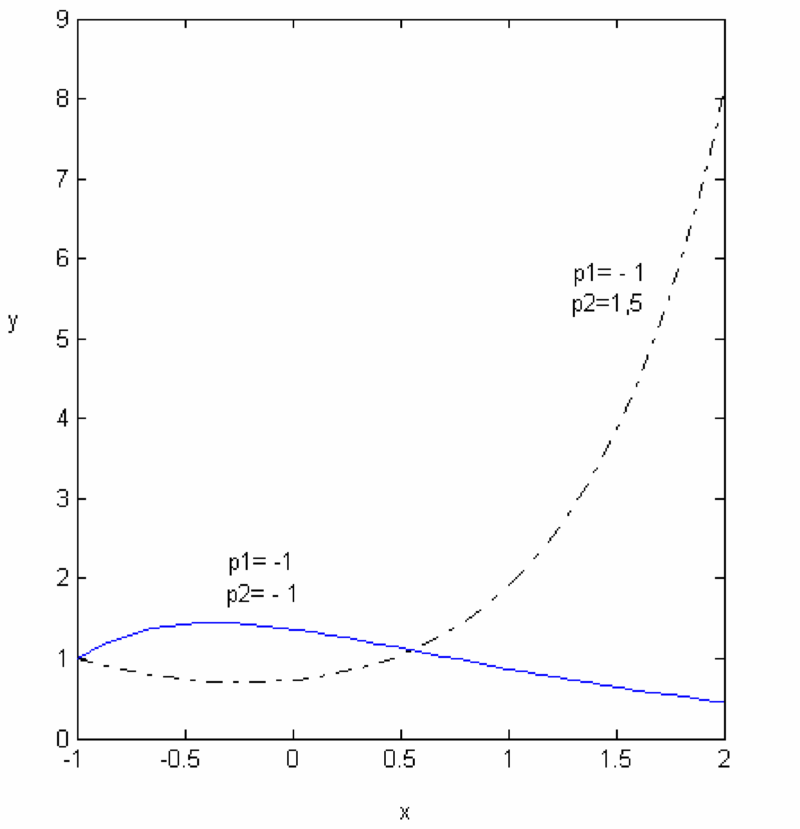
Rezolvare.
- Se creează fișierul funcție **g.m**:

```
function yprim=g(x,y,p)
yprim=-p*y;
```

- Se apelează funcția `ode45` pentru integrare numerică succesiv pentru două seturi de valori ale parametrilor p1 și p2, astfel:

```
[x,y]=ode45(@g,[-1 2],1,[],-1,1.5)
[x,y]=ode45(@g,[-1 2],1,[],-1,-1)
```

Se reprezintă grafic soluțiile numerice pentru cele două cazuri.



B. Ecuații diferențiale de ordin superior. Sisteme de ecuații diferențiale

Utilizarea funcțiilor Matlab pentru rezolvarea numerică a ecuațiilor diferențiale de ordin superior şi a sistemelor de ecuații diferențiale presupune aranjarea preliminară a acestora într-o formă recursivă reductibilă la ecuații diferențiale de ordinul întâi. Astfel, *o ecuație diferențială de ordin superior este echivalentă cu un sitem de ecuații diferențiale de ordinul întâi* cuplate recursiv. Pentru o ecuație de ordinul n, aceasta se exprimă notând derivatele succesive ale funcției cu y_1, y_2, ..., y_n, astfel:

Forma standard		Forma echivalentă
$y^{(n)} = f(x, y, y', y'', ..., y^{(n-1)})$	$\Leftrightarrow$ $\Uparrow$ *Cu notațiile:* $\Downarrow$	$\begin{cases} y_1' = y^{(n)} = f(x, y_n, y_{n-1}, ..., y_1) \\ y_2' = y_1 \\ ... \\ y_{n-2}' = y_{n-3} \\ y_{n-1}' = y_{n-2} \end{cases}$

$$\begin{cases} y_1(x) = y^{(n-1)} \\ y_2(x) = y^{(n-2)} \\ ... \\ y_{n-2}(x) = y'' \\ y_{n-1}(x) = y' \\ y_n(x) = y \end{cases}$$

Exemple.

(1) Să se rezolve ecuația diferențială de ordinul cinci, $y^{(5)} = 0$ pe intervalul [-1,2], cu următoarele condiții inițiale:

$$\begin{cases} y(-1) = 1 \\ y'(-1) = 0 \\ y''(-1) = -1 \\ y'''(-1) = 0 \\ y^{(4)}(-1) = 1 \end{cases}$$

Rezolvare. Ecuația este deja în forma standard şi este echivalentă cu următorul sistem de ecuații diferențiale de ordinul întâi:

$$\begin{cases} y_1' = 0 \\ y_2' = y_1 \\ y_3' = y_2 \\ y_4' = y_3 \\ y_5' = y_4 \end{cases},$$

în care s-au introdus notațiile y_1, y_2, y_3, y_4.

Se crează fişierul funcție **ecdif5.m** ce descrie sistemul de ecuații diferențiale de ordinul întâi echivalent:

```
function yprim=ecdif5(x,y)
yprim=zeros(5,1); %vectorul derivatelor
yprim(1)=0;
yprim(2)=y(1);
yprim(3)=y(2);
yprim(4)=y(3);
yprim(5)=y(4);
```

Se apelează funcția `ode` cu argumentele necesare pentru integrarea numerică a ecuației diferențiale ecdif5:

```
[x,y]=ode23(@ecdif5,[-1 2],[1 0 -1 0 1])
```

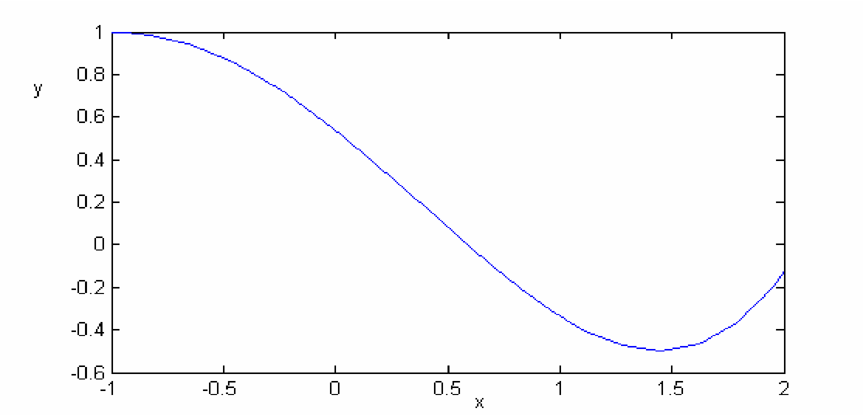
Funcția returnează o structura de date de forma:

```
Name      Size          Bytes  Class

  x       24x1           192   double array
  y       24x5           960   double array
```

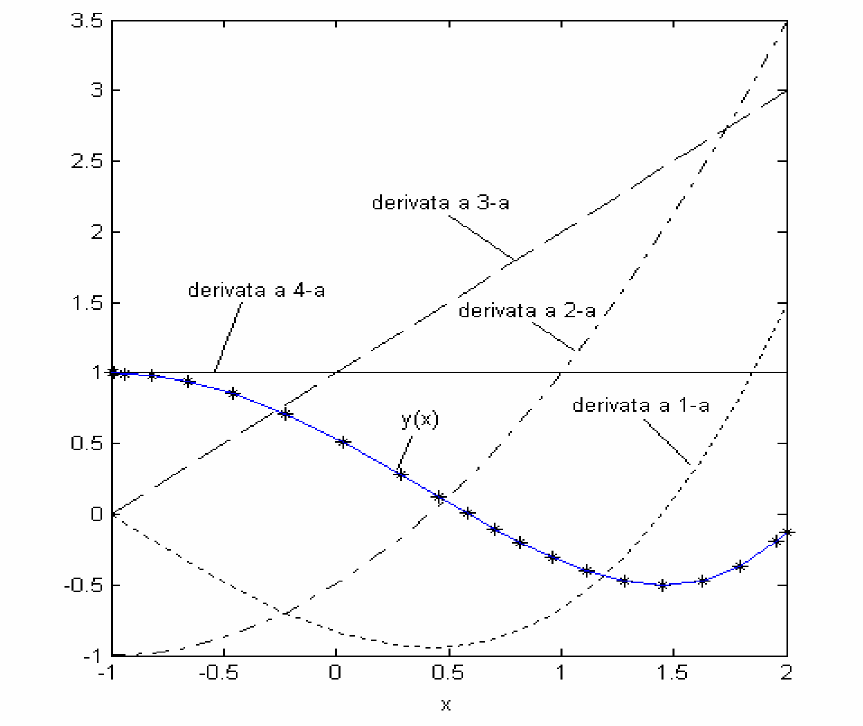
Rezultatele numerice returnate sunt:

Argument	Componentele soluției (derivatele succesive- pe coloane)				
x =	y =				
-1.0000	**1.0000**	**0**	**-1.0000**	**0**	**1.0000**
-0.9999	1.0000	0.0001	-1.0000	-0.0001	1.0000
-0.9995	1.0000	0.0005	-1.0000	-0.0005	1.0000
-0.9975	1.0000	0.0025	-1.0000	-0.0025	1.0000
-0.9875	1.0000	0.0125	-0.9999	-0.0125	0.9999
-0.9375	1.0000	0.0625	-0.9980	-0.0624	0.9980
-0.8222	1.0000	0.1778	-0.9842	-0.1769	0.9842
-0.6590	1.0000	0.3410	-0.9418	-0.3344	0.9424
-0.4572	1.0000	0.5428	-0.8527	-0.5162	0.8562
-0.2239	1.0000	0.7761	-0.6989	-0.6982	0.7138
0.0340	1.0000	1.0340	-0.4654	-0.8497	0.5127
0.2909	1.0000	1.2909	-0.1668	-0.9324	0.2819
0.4573	1.0000	1.4573	0.0618	-0.9415	0.1254
0.5811	1.0000	1.5811	0.2500	-0.9223	0.0098
0.7049	1.0000	1.7049	0.4534	-0.8789	-0.1020
0.8187	1.0000	1.8187	0.6538	-0.8161	-0.1986
0.9576	1.0000	1.9576	0.9161	-0.7073	-0.3049
1.1140	1.0000	2.1140	1.2346	-0.5394	-0.4031
1.2814	1.0000	2.2814	1.6024	-0.3024	-0.4744
1.4536	1.0000	2.4536	2.0101	0.0083	-0.5007
1.6248	1.0000	2.6248	2.4448	0.3892	-0.4678
1.7922	1.0000	2.7922	2.8981	0.8359	-0.3664
1.9525	1.0000	2.9525	3.3585	1.3370	-0.1933
2.0000	1.0000	3.0000	3.5000	1.5000	-0.1259



Tabloul grafic al tuturor componentelor succesive care intervin în rezolvarea ecuației diferențiale respectiv derivatele succesive care se integrează numeric până la obținerea soluției finale $y(x)$ este prezentat mai jos. Acesta se obține cu o instrucțiune de trasare grafică de forma următoare:

```
>> plot(x,y(:,1),'-k',x,y(:,2),'--k',x,y(:,3),'-.k',x, ...
y(:,4),':k',x,y(:,5),'*k',x,y(:,5))
```



(2) Să se rezolve ecuația diferențială $y^{(4)} - 2y'' - 5y = 0$ pe intervalul [-2,-1], cu următoarele condiții inițiale:

$$\begin{cases} y(-2) = -2 \\ y'(-2) = 0 \\ y''(-2) = 0 \\ y'''(-2) = 2 \end{cases}$$

Rezolvare. Se aduce ecuația la forma standard exprimând derivata de ordinul cel mai mare ca funcție de celelalte derivate şi eventual variabila independentă:

$$y^{(4)} = 2y'' + 5y\ ,$$

ceea ce se exprimă echivalent prin sistemul de ecuații diferențiale:

$$\begin{cases} y^{(4)} = 2y'' + 5y \\ y''' = y_1 \\ y'' = y_2 \\ y' = y_3 \end{cases},\ \text{respectiv}\ \begin{cases} y_1' = 2y_2 + 5y \\ y_2' = y_1 \\ y_3' = y_2 \\ y' = y_3 \end{cases},\ \text{unde}\ y = y_4 .$$

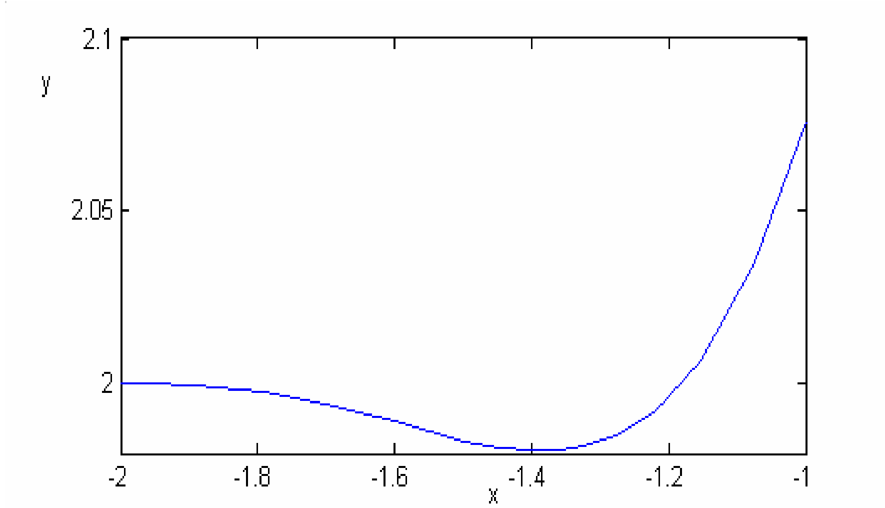
Se scrie fişierul funcție **ecdif4.m**:

```
function yprim=ecdif4(x,y)
yprim=zeros(4,1); %vectorul derivatelor
yprim(1)=2*y(2)+5*y(4);
yprim(2)=y(1);
yprim(3)=y(2);
yprim(4)=y(3);
```

Scriind succesiv în linie de comandă instrucțiunile:

```
>> [x,y]=ode23(@ecdif4,[-2 -1],[-2 0 0 2])
>> plot(x,y(:,4))
```

se obține graficul funcției soluție y(x) obținută prin integrare numerică în condițiile date.



(3) Să se găsească soluția ecuației diferențiale $xy''' + y'' = 1 + x$ în intervalul [-2,0], pentru următorul set de condiții inițiale:

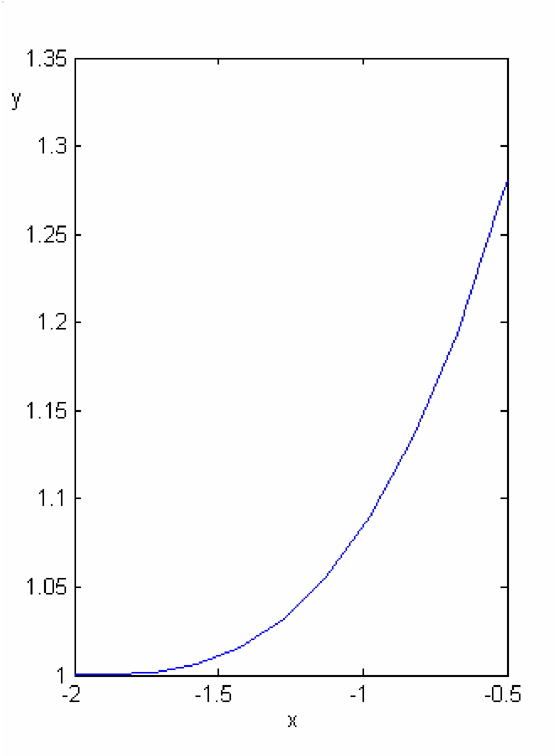
$$\begin{cases} y(-2) = 0 \\ y'(-2) = 0 \\ y''(-2) = 1 \end{cases}$$

Rezolvare. Fişierul **ecdif3.m** implementează sistemul echivalent al ecuației diferențiale:

```
function yprim=ecdif3(x,y)
yprim=zeros(3,1); %vectorul derivatelor
yprim(1)=1+1/x-(1/x)*y(1);
yprim(2)=y(1);
yprim(3)=y(2);
```

Integrarea numerică şi reprezentarea grafică a soluției se obțin cu succesiunea de instrucțiuni:

```
>> [x,y]=ode23(@ecdif3,[-2 -0.5],[0 0 1])
>> plot(x,y(:,3))
```



(4) Să se integreze ecuaţia diferenţială de ordinul doi $(1+x^2)y''+y'^2+1=0$ pe intervalul [-1 2] pentru condiţiile iniţiale:

$$\begin{cases} y(-1)=1 \\ y'(-1)=0 \end{cases}$$

Rezolvare. Se parcurg paşii necesari cunoscuţi.

Fişierul funcţie **ecdif2.m**:

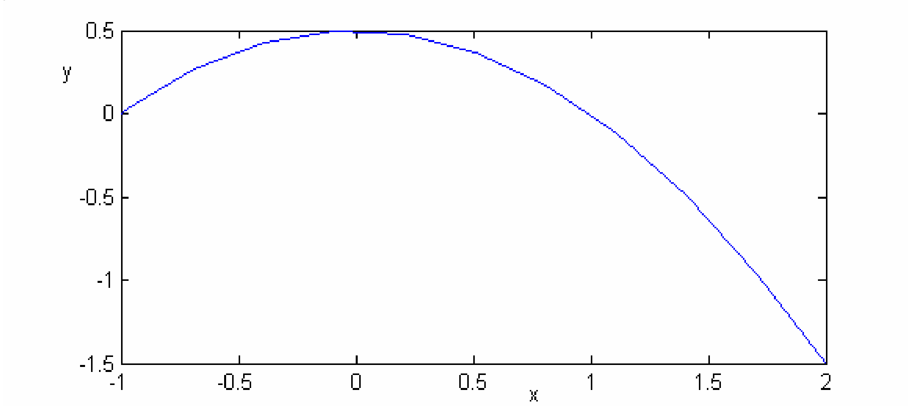
```
function yprim=ecdif2(x,y)
yprim=zeros(2,1); %vectorul derivatelor
yprim(1)=-1/(1+x^2)-1/(1+x^2)*y(1)^2;
yprim(2)=y(1);
```

Apelul funcţiei de integrare numerică pentru condiţiile date este:

```
>> [x,y]=ode23(@ecdif2,[-1 2],[1 0])
```

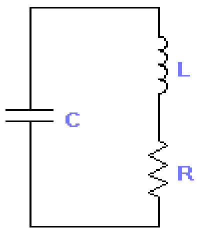
Reprezentarea grafică a soluției:

```
>> plot(x,y(:,2))
```



Problemă.

Fie circuitul electric format prin conectarea unui condensator C la bornele unui circuit serie compus dintr-o bobină L şi o rezistență R, conform schemei de mai jos:



Să se determine evoluția tensiunii la bornele condensatorului $u(t)$ şi a curentului $i(t)$ în circuitul dat pe intervalul de timp [0, 5T], unde T este perioada de oscilație proprie a circuitului, pentru următoarele condiții inițiale:

$$\begin{cases} u(0) = 40V \\ u'(0) = 0 \end{cases}.$$

Valorile elementelor de circuit L, C şi R vor fi considerați parametrii şi transferați funcției ca atare.

Rezolvare.

Problema constă în rezolvarea ecuației diferențiale circuitului LC paralel real (bobina are o rezistență activă proprie R). Astfel, în regim tranzitoriu ecuația căderilor de tensiune în circuit este:

$LCu'' + RCu' + u = 0$, iar curentul este dat de $i = -Cu'$.

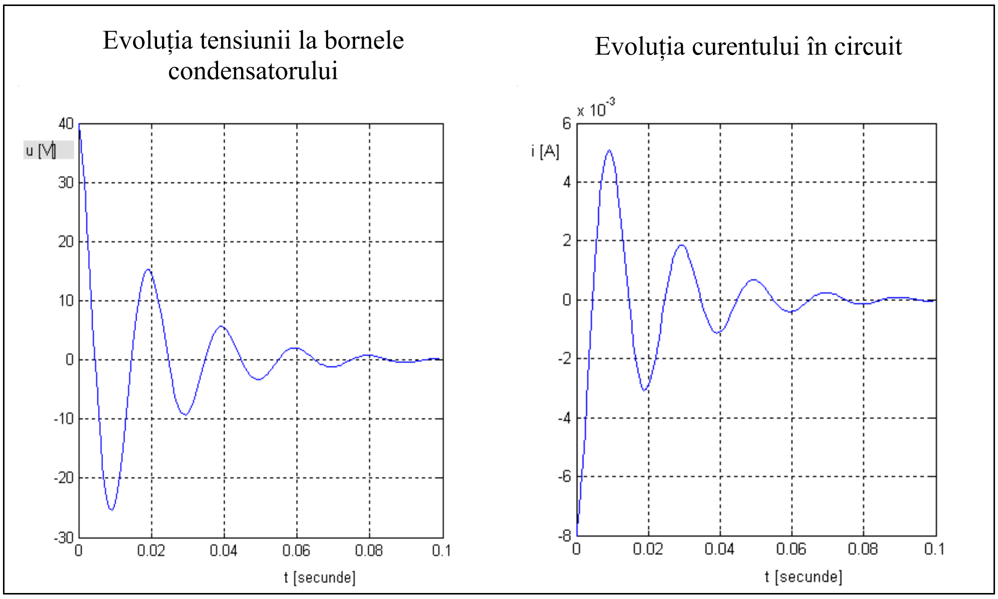
- Se scrie fişierul funcție al sistemului echivalent pe care îl denumim **circuit.m**:

```
function uprim=circuit(t,u,L,C,R)
uprim=zeros(2,1); %vectorul derivatelor
uprim(1)=(-R/L)*u(1)-(1/(L*C))*u(2);
uprim(2)=u(1);
```

- Se scrie un program principal care rezolvă cerințele problemei, de pildă pentru valorile L=50μH, C=200μF, R=5Ω, astfel:

```
%program principal
clear
u0=40
up0=0
L=50*10^(-3)
C=200*10^(-6)
R=5
T=2*pi*sqrt(L*C);
tf=5*T;
[t,u]=ode45(@circuit,[0 tf],[u0 up0],[],L,C,R)
figure(1)
plot(t,u(:,1));
grid
i=-C*u(:,1)
figure(2)
plot(t,i);
grid
```

Rezultatul grafic este reprezentat mai jos.



<u>*Exerciții propuse spre rezolvare*</u>.

Să se rezolve numeric pe intervale şi pentru condiții inițiale la alegere următoarele ecuații diferențiale:

1. $y'' = 1 + \mathrm{tg}^2 x$
2. $y''' = \ln x$
3. $y'' = \dfrac{y'}{x} + \dfrac{x^2}{y'}$
4. $yy'' - y'^2 - 2yy' \ln y = 0$
5. $1 + y'^2 = 2yy'$
6. $3y'y'' = y + y'^3 + 1$
7. $y^{(4)} - 2y''' + y'' = x + e^x$
8. $y^{(4)} + y''' = \cos 4x$