Гранична задача за обикновени линейни диференциални уравнения от 2 ред

Дадено е диференциалното уравнение от вида:

$$a(x)y''(x)+b(x)y'(x)+c(x)y(x)=f(x)\ , \tag{1}$$

където $a \le x \le b$, $a(x) \neq 0$, а граничните условия са зададени за двата края на интервала на независимата променлива *x* и са записани в един от двата вида:

I тип $y(a)=\gamma_1$ и $y(b)=\gamma_2$ (2)

II тип $\alpha_1 y(a)+\beta_1 y'(a)=\gamma_1$ $\alpha_1 y(a)+\beta_1 y'(a)=\gamma_1$ и $\alpha_2 y(b)+\beta_2 y'(b)=\gamma_2$ (3)

Функциите $a(x), b(x), c(x), f(x)$ имат краен брой точки на прекъсване в интервала (a,b) , а константите $\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2$ не са едновременно равно на нула.

Най-често решението се търси по „**метода на стрелбата**“ или по метод **преобразуващ диференциалната задача в система от линейни алгебрични уравнения**, например чрез заменяне на производните чрез крайни разлики от стойности на функцията в избрани точки, по метода на крайните елементи и др.

Преписваме уравнение (1) в по компактния вид:

$$y''=p(x)y'+q(x)y+r(x) \tag{4}$$

Тук функциите $p(x)=-\frac{b(x)}{a(x)},\ q(x)=-\frac{c(x)}{a(x)},\ r(x)=\frac{f(x)}{a(x)}$.

Метод на стрелбата

Ще разгледаме подробно случая за гранични условия от I тип.

Решават се две задачи на Коши:

$$y_1''=p(x)y_1'+q(x)y_1+r(x) \tag{5}$$

с начално условие

$$y_1(a)=\gamma_1,\ y_1'=0 \tag{6}$$

Намираме решението за $y_1(x)$ и от втората за $y_2(x)$

$$y_2''=p(x)y_2'+q(x)y_2 \tag{7}$$

с начално условие

$$y_2(a)=0,\ y_2'=1 \tag{8}$$

Решението на граничната задача се дава с израза

$$y(x) = y_1(x) + \frac{\gamma_2 - y_1(b)}{y_2(b)} y_2(x) \qquad (9)$$

Решаването на задачи (5)-(8) може да се направи посредством вградените функции на Matlab – ODE45, ODE23 и т.н., като предварително *диференциалните уравнения от* II ред *се сведат до система от две обикновени диференциални уравнения* от I ред.

АЛГОРИТЪМ за метода на стрелбата

I стъпка: Свеждаме линейното диференциално уравнение от II ред (5)-(6) към система от I ред чрез полагането $y_1' = u,\ \mathrm{y}_1 = v$. Получаваме системата:

$$u' = p(x)u + q(x)v + r(x), \qquad (10)$$

$$v' = u \qquad (11)$$

с начални условия

$$v(a) = \gamma_1 \text{ и } u(a) = 0 \qquad (12)$$

Свеждаме линейното диференциално уравнение от II ред (7)-(8) към система от I ред чрез полагането $y_1' = u,\ \mathrm{y}_1 = v$. Получаваме системата:

$$u' = p(x)u + q(x)v \qquad (13)$$

$$v' = u \qquad (14)$$

с начални условия

$$v(a) = 0 \text{ и } u(a) = 1 \qquad (15)$$

II стъпка: В Matlab функции fun1(x,z), fun2(x,z), които **връщат вектор-стълб** се описват десните страни на системи (10)-(11) и (13)-(14)

```
function yp=fun1(x,z)
yp=zeros(2,1);          % задава yp като вектор стълб
u=z(1); v=z(2);         %присвоява данните от z в променливите u и v
yp(1)=.....             % описва уравнение (10)
yp(2)=.....             % описва уравнение (11)
end
```

Аналогично

```
function yp=fun2(x,z)
yp=zeros(2,1);          % задава yp като вектор стълб
u=z(1); v=z(2);         %присвоява данните от z в променливите u и v
yp(1)=.....             % описва уравнение (13)
yp(2)=.....             % описва уравнение (14)
end
```

III стъпка: В Matlab функция myfun се решават системите посредством вградената адаптивна процедура на Рунге-Кута от 4-5 ред ODE45…

Примерна част от код на mufun

```
function myfun
clc;
a=input('Задай min стойност на независимата променлива x a =');
b=input('Задай max стойност на независимата променлива x b =');
n=input('В колко точки да се пресметне решението n = ');
g1=input('Задай начално условие при x=a  y(a)=');
g2=input('Задай начално условие при x=b  y(b)=');
h=(b-a)/n;
z0=[0;g1];                          % начални условия (12) вектор-стълб z0
[x,z]=ode45(@fun1,[a:h:b],z0);
y1=z(:,2);                          %помества решенията в y1
z0=[1;0];                           % начални условия (15) вектор-стълб z0
[x,z]=ode45(@fun2,[a:h:b],z0);
y2=z(:,2);                          %помества решенията в y2
y=y1+(g2-y1(n))./y2(n)*y2;          %решение на граничната задача
plot(x,y,'-*');                     %графично представяне на решението
end
```

Като втори метод ще разгледаме метода на мрежите.

Метод на мрежите

Представяме производните чрез следните триточкови шаблони за стъпка $h = x_{i+1} - x_i$:

$$y'(x_i) = \frac{y(x_{i+1}) - y(x_{i-1})}{x_{i+1} - x_{i-1}} = \frac{y_{i+1} - y_{i-1}}{2h} + \mathrm{O}(h^2), \tag{16}$$

$$y''(x_i) = \frac{y(x_{i+1}) - 2y(x_i) + y(x_{i-1})}{(x_{i+1} - x_i)(x_i - x_{i-1})} = \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} + \mathrm{O}(h^2) \tag{17}$$

$y'(x_0) = y_0' = \frac{-3y_0 + 4y_1 - y_2}{2h} + \mathrm{O}(h^2)$ за левия край на интервала $x_0 = a$ (18)

$y'(x_n) = y_n' = \frac{3y_n - 4y_{n-1} + y_{n-2}}{2h} + \mathrm{O}(h^2)$ за десния край на интервала $x_n = b$ (19)

Заместваме в уравнение (4) и в гранични условия (2) или (3), след което групирайки членовете пред неизвестните y_{i-1}, y_i, y_{i+1} получаваме следната тридиагонална алгебрична система уравнения:

$$-\left(1 + \frac{h}{2}p(x_i)\right)y_{i-1} + \left(2 + h^2 q(x_i)\right)y_i - \left(1 - \frac{h}{2}p(x_i)\right)y_{i+1} = -h^2 r(x_i) \tag{20}$$

за $i = 1, 2, \cdots, n-1$. Тази система се смята , че има устойчиво решение, ако стъпката е подбраната така, че $h < \frac{2}{\max_{a \le x \le b} |p(x)|}$.

Алгоритъм за мрежов метод за гранична задача от I тип на линейно ОДУ II ред

Стъпка I: Съставяме три файл функции на Matlab, описващи *p(x)*, *q(x)* и *r(x)*.

```
function  y = p(x)
  y=  …. % дясната част на p(x)
end

function  y =q (x)
  y=  …. % дясната част на q(x)
end

function  y = r(x)
  y=  …. % дясната част на r(x)
end
```

Стъпка II: Съставяме матрицата от коефициентите на линейната тридиагонална система и вектора на свободния стълб.

Стъпка III: Решаваме системата

Примерна част от код на mufun (стъпка 2 и 3)

```
function myfun
clc;
a=input('Задай min стойност на независимата променлива x a =');
b=input('Задай max стойност на независимата променлива x b =');
n=input('В колко точки да се пресметне решението n = ');
g1=input('Задай начално условие при x=a  y(a)=');
g2=input('Задай начално условие при x=b  y(b)=');
h=(b-a)/n; h2=h/2;hkw=h*h;
% Изчисляване на коефициентите
for i=1:n
    x(i)=a+i*h;
end
A=zeros(n); B=zeros(1,n);
for i=1:n
    Q(i)=2*hkw*q(x(i));
    PL(i)=-1-h2*p(x(i));
    PU(i)= -1+h2*p(x(i));
    R(i)=-hkw*r(x(i));
end
% Получаваме тридиагоналната матрица
A=diag(PL(2:n),-1)+diag(Q)+diag(PU(1:n-1),1);
```

```
B(1,1)=-PL(1)*g1;                % ляво гранично условие
B(1,n)=-PL(n)*g2;                % дясно гранично условие
B=B+R; B=B'                      % превръщаме във вектор стълб
% Решаваме системата
Y=A^-1*B;
plot(x,Y,'-*');
end
```