Лабораторна робота № 16.

Тема. Розв’язання математичних моделей, створених на основі звичайних диференціальних рівнянь.

Мета роботи: навчитись розв’язувати крайову задачу та представляти розв’язок у графічному вигляді у системі **MATLAB**.

Теоретичний мінімум

Важливим елементом завдань, які передбачають дослідження математичних моделей, що складаються з звичайних диференціальних рівнянь, є додаткові умови, які необхідні для реалізації чисельного алгоритму розв’язання. Стосовно звичайних диференціальних рівнянь розрізняють два види завдань: завдання з початковими умовами (задача Коші) і завдання з граничними умовами (крайова задача).

Таким чином, коли розглядається крайова задача, потрібно знайти розв’язок звичайного диференціального рівняння з додатковими умовами, які задаються при декількох різних значеннях незалежної змінної [20]. Наприклад, для звичайного диференціального рівняння другого порядку

$$\frac{d^2u}{dx^2} = f\left(x, u, \frac{\partial u}{\partial x}\right) \qquad (1)$$

ці умови задаються у двох різних точках (різних значеннях незалежної змінної) $x = a$ та $x = b$:

$$u(a) = A,\ u(b) = B. \qquad (2)$$

Оскільки ці точки визначають границі області $a < x < b$, в якій зазвичай і відшукується розв’язок, то визначені в них додаткові умови називають граничними або краєвими. У інженерній практиці, наприклад, таке завдання може бути пов'язане з розрахунком прогину стержню при заданому способі закріплення його кінців.

Для побудови чисельного розв’язку можна застосувати наступні два способа розв’язання крайової задачі:

- метод скінченних різниць;
- метод приведення до задачі Коші шляхом заміни додаткових умов.

Наприклад, при розв’язанні краєвої задачи методом пристрілювання використовується приведення до задачі Коші.

Якщо задана крайова умова, то в методі пристрілювання вона замінюється умовою Коші для шуканої функції вихідного рівняння (1):

$$u(a) = A,\ \left.\frac{du}{dx}\right|_{x=a} = k = tg\theta, \qquad (3)$$

де $u(a)$ – точка, яка є початком інтегральної кривоїдиференціального рівняння, тобто функції $u(x)$, а θ – кут нахилу дотичної до цієї кривої в початковій точці $x = a$.

Таким чином, розв’язок задачі Коші буде залежити від початкової умови $\left.\frac{du}{dx}\right|_{x=a} = tg\theta$. Необхідно підібрати значення кута θ, при якому інтегральна крива $u(x)$ в точці b дасть збігаючий з умовою $u(b) = B$ результат. Якщо ця умова буде виконана, то розв’язок задачі Коші збігається з розв’язком крайової задачі [51].

Відносно назви "метод пристрілювання" зауважимо, що ця назва витікає з процедури визначення (як би "пристрілки") величини кута нахилу кривої $u(x)$ в початковій точці.

У випадку, коли звичайне диференціальне рівняння другого порядку є лінійним, тобто має вигляд

$$\frac{d^2u}{dx^2} = f_1(x)\frac{du}{dx} + f_2(x)u + f_3(x),$$

і задані граничні умови (2), то пошук розв’язку методом пристрілювання істотно спрощується. Дійсно, якщо визначити дві шуканих "пристрілювальних" величини кутів θ_1 і θ_2, отримуємо два розв’язки $u_1(x)$ і $u_2(x)$. У випадку, коли $u_1(b) = B_1$ і $u_2(b) = B_2$, причому $B_1 \neq B_2$, то розв’язком крайової задачі буде лінійна комбінація цих двох розв’язків, а саме:

$$u(x) = \frac{1}{B_1 - B_2}\left[(B - B_2)u_1(x) + (B_1 - B)u_2(x)\right].$$

Для звичайного диференціального рівняння 2-го порядку число граничних умов повинне дорівнювати 2 (тобто для звичайного диференціального рівняння порядку $n \geq 2$ число граничних умов повинне дорівнюватись порядку рівняння). Для рівняння довільного порядку n граничні умови (в кількості n) можуть включати значення на кінцях відрізка невідомої функції і її похідних до $(n-1)$-го порядку включно.

У випадку чисельного розв’язання крайової задачі для звичайного диференціального рівняння $(n-2)$-го порядку вихідне рівняння звичайною процедурою зводять до системи з n рівнянь 1-го порядку. Аналогічна процедура застосовується і до граничних умов.

Вправа 1. Чисельне розв’язання крайової задачі з використанням команди **bvp4c**.

У системі **MATLAB** існує команда **bvp4c**, яка дозволяэ розв’язати крайову задачу для системи звичайних диференціальних рівнянь першого порядку виду $y' = f(\bar{x}, \bar{y})$, де $\bar{y}$ – вектор, $\bar{y}'$ – вектор, що складається з похідних від компонентів вектора $\bar{y}$, $f(\bar{x}, \bar{y})$ – функція від векторів $\bar{x}$ та $\bar{y}$.

Розв'язок шукається у формі сіткової функції: відрізок (a, b) ділиться точками $a = x_1 < x_2 < ... < x_n = b$ на частини (не обов'язково рівні) і кожній точці x_i ставиться у відповідність значення y_i. Виходячи з початкових значень x_i і y_i шляхом апроксимації похідної $y' = \frac{dy}{dx}$ в кожній точці сітки відношенням $\frac{\Delta y}{\Delta x}$, будується система рівнянь, з якої знаходяться значення y. В процесі розв'язання задачі сітка може перебудовуватися (зокрема, згущуватися).

Початкові значення x_i і y_i задаються за допомогою команди **bvpinit**: **solinit = bvpinit(xinit,yinit)**, де :

- **xinit** – вектор-рядок x_i-координат вузлових точок (**a=xinit(1) <xinit(2) <...<xinit(n)=b**);
- **yinit** – гіпотетичні значення (тобто значення, які задаються довільним чином) для **y(i);** ці значення можуть задаватися в одній з двох форм:
    - векторній формі – кожна компонент вектора **yinit(i)** формується як гіпотетичний розв'язок для всіх точок сітки, тобто **y(i, :)=yinit(i)**;
    - функціональній формі – задається у вигляді **y=guess(x)**, де **x** – будь-яка точка відрізку **[a, b]**, **y** – вектор, довжина якого дорівнює порядку системи диференціальних рівнянь; для кожної точки сітки **x(i)** обчислюється вектор гіпотетичного розв'язку **y(i, :)=guess(x(i))**;
- **solinit** – структура, яка формується при вказаному вище способі звернення до команди **bvpinit** у формі поелементного заповнення двох векторів **x**, **y**:
    - **solinit.x = xinit**
    - **solinit.y = y(i, :)**

Розв'язок крайової задачі можна знайти, якщо використати команду **bvp4c**, для якої існує проста форма звернення:

sol=bvp4c(odefun,bcfun,solinit)

де:

- **odefun** – функція, що формує (обчислює) вектор правих частин;
- **bcfun** – функція, що обчислює вектор граничних умов; аргументами функції **bcfun** є вектори **ya** і **yb** – вектори розв'язку **y** в точках **a** і **b**; двома компонентами вектора граничних умов **bcfun** є вирази, які обертаються в нуль в точках **a** і **b** відповідно;

- **solinit** – вихідна структура команди **bvpinit**, яка містить в собі початкові умови задачі **xinit,yinit**;
- **sol** – аналогічна структура, що містить розв’язок крайової задачі; окрім параметрів **sol.x** і **sol.y** структура **sol** має параметр **sol.yp**, у якому містяться значення похідної розв’язку **(sol.y)** в точках **sol.x**.

Приклад 1. Розв’язується диференціальне рівняння другого порядку $y'' + y = 0$ (рівняння вільних коливань) з граничними умовами $y(0) = 0,\ y(\pi) = 1$. Треба знайти розв’язок крайової задачі за допомогою засобів системи **MATLAB**.

Вихідне диференціальне рівняння приводиться до системи рівнянь:

$$\begin{cases} y_1{}' = y_2 \\ y_2{}'' = -y_1 \end{cases},$$

Інформацію про яку поміщаємо в робочий каталог у вигляді **m**-файлу, що містить також функцію, яка формує (обчислює) праві частини:

```
function dydx=kr1(x,y)
dydx=[y(2); –y(1)];
```

і **m**-файлу, що містить функцію, яка задає граничні умови:

```
function res=bord(ya,yb)
res=[ya(1); yb(1)–1];
```

За допомогою команди **linspace** визначаємо початкові значення змінної x шляхом завдання п'яти рівновіддалених точок на інтервалі $[0\ \ \pi]$:

```
>> xinit=linspace(0,pi,5)
xinit =
      0    0.7854    1.5708    2.3562    3.1416
```

Початкові наближення шуканого вектора розв’язку задачі можна задати вектором **yinit**, розмір якого не перевищує величини, що визначає порядок диференціального рівняння. Вибір структури початкового наближення залежить від дослідника.

Скористаємось допоміжною функцією, яка формує компоненти вектора розв’язку **yinit**. Наприклад, сформуємо **m**-файл **sincos**:

```
function sc=sincos(x)
sc=[sin(x); cos(x)];
```

За допомогою команди **bvpinit** можна утворити структуру **solinit, (**яка містить початкові умови**)**:

```
>> solinit=bvpinit(xinit,@sincos)
solinit =
   x: [0 0.7854 1.5708 2.3562 3.1416]
   y: [2x5 double]
>> solinit.x
ans =
      0    0.7854    1.5708    2.3562    3.1416
```

```
>> solinit.y
ans =
         0    0.7071    1.0000    0.7071    0.0000
    1.0000    0.7071    0.0000   –0.7071   -1.0000
```

Для розв’язку задачі достатньо виконати команду **bvp4c**:

```
>> sol=bvp4c(@kr1,@bord,solinit)
sol =
         x: [1x13 double]
         y: [2x13 double]
        yp: [2x13 double]
    solver: 'bvp4c'
```

Для отримання чисельних значень шуканого розв’язку, треба послідовно виконати наступні команди:

```
>> sol.x
ans =
  Columns 1 through 7
         0    0.1963    0.3927    0.7854    1.1781    1.3744    1.5708
  Columns 8 through 13
    1.7671    1.9635    2.3562    2.7489    2.9452    3.1416
>> sol.y
ans =
  1.0e+004 *
  Columns 1 through 7
         0    0.3570    0.7003    1.2940    1.6908    1.7949    1.8301
    1.8301       1.7949       1.6908       1.2941       0.7004       0.3571
0.0001
  Columns 8 through 13
    1.7949    1.6908    1.2941    0.7004    0.3571    0.0001
   –0.3570   –0.7003   –1.2940   –1.6907   –1.7949   –1.8301
>> sol.yp
ans =
  1.0e+004 *
  Columns 1 through 7
    1.8301       1.7949       1.6908       1.2941       0.7004       0.3571
0.0001
         0    –0.3570    –0.7003    –1.2940    –1.6908    –1.7949    –
1.8301
  Columns 8 through 13
   –0.3570   –0.7003   –1.2940   –1.6907   –1.7949   –1.8301
   –1.7949   –1.6908   –1.2941   –0.7004   –0.3571   –0.0001
```

Якщо спробувати побудувати розв’язок крайової задачі у вигляді графіка за допомогою команди **plot(sol.x, sol.y(l, :))**, то зауважимо, що результат вийде не дуже гладкий – дуже мало точок містить вектор **sol.x**.

Для побудови детального графіку треба скористатися додатковою командою **deval**. Вона використовує інформацію, яка міститься в структурі **sol (**масив розв'язків**)**, для побудови інтерполяційного сплайну для заданого вектора пробних точок **xx**. Вектор **xx** може містити скільки завгодно точок для забезпечення необхідної гладкості результату. Синтаксис цієї команди має вигляд **yy=deval(sol,xx)**, де:

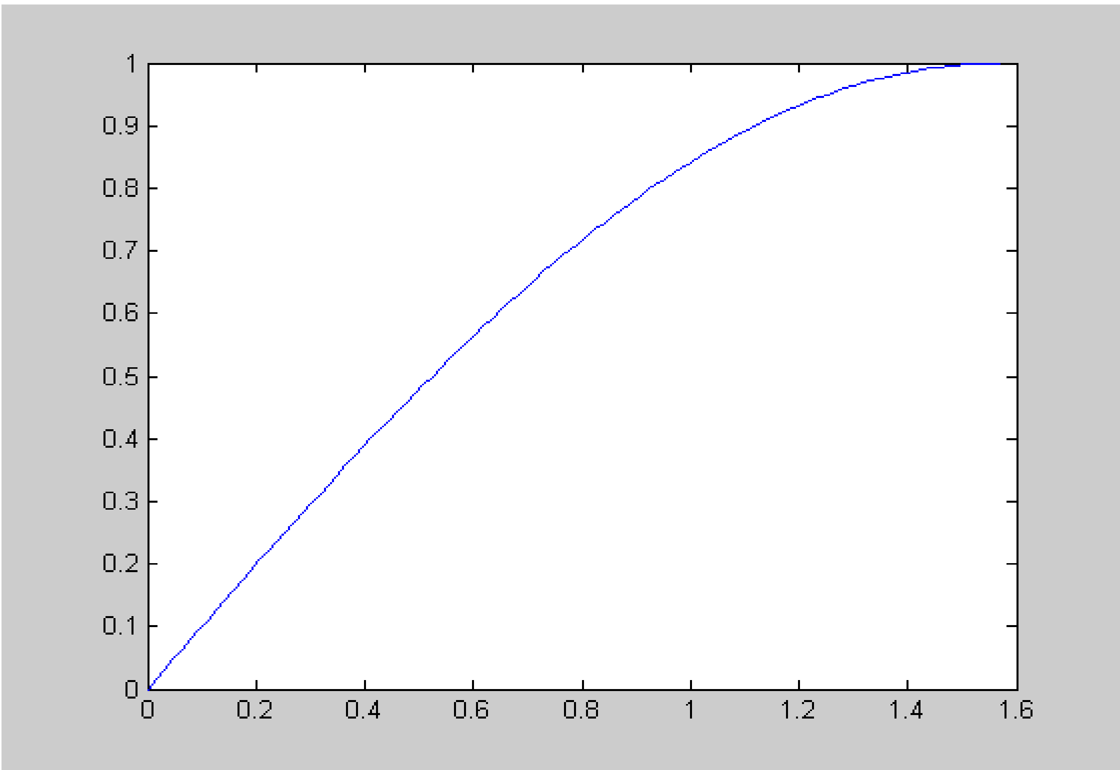
- **sol** – параметр, що містить отримані результати розв'язання крайової задачі, які одержуються за допомогою команди **bvp4c**;
- **xx** – вектор пробних точок;
- **yy** – вектор значень інтерполяційної сплайн-функції в пробних точках.

Зауважимо, що кожен компонент вектору **yy** насправді є двокомпонентним вектором (подібно до елементів **sol.y**), тому для побудови графіка достатньо використати команду **plot(xx, yy(1, :)),** в якій присутня інформація про перший стовпець (розв'язок крайової задачі) і відсутня інформація про другий стовпець (похідна розв'язку крайової задачі)**.**

Повний перелік команд побудови шуканого графіку розв'язку крайової задачі наведений нижче:

```
>> xx=linspace(0,pi/2); yy=deval(sol,xx);
>> plot(xx,yy(1,:))
```

Графічний результат обчислень представлений на мал. 15.1.



Мал. 15.1 Графік розв'язку диференціального рівняння на відрізку $[0, \pi/2]$**.**

Вправа 2. Розв’язання крайової задачі на основі додаткової інформації про якість наближення.

Повна форма звернення до команди **bvp4c: sol=bvp4c(odefun,bcfun,solinit,OPTIONS,PI,P2...)** дозволяє:

- враховувати додаткові умови, які використовуються при розв’язуванні;
- використовувати додаткові параметри у командах **odefun** і **bcfun,** щоб задати координати, в яких буде обчислюватись результат;
- знаходити значення невідомих параметрів (наприклад, власних чисел).

В такій формі звернення вихідними аргументами є:

• **OPTIONS** – параметр, що дозволяє задавати додаткову інформацію про чисельний процес;

• **PI**, **P2** ... – додаткові аргументи для обчислення **odefun** і **bcfun**.

Аргумент **OPTIONS** дозволяє відобразити відповідні параметри, які використовуються для управління якістю наближення шуканого розв’язку. Нагадаємо ці основні параметри. Всі вони формуються при використанні команди **bvpset**, вхідними параметрами якої є впорядкована послідовність пар виду <'параметр', значення> (назва будь-якого з параметрів повинна бути поміщена в апострофи).

Параметр **RelTol** задає допустиму відносну похибку обчислення (за умовчанням величина параметру дорівнює **1e-3)**.

Параметр **AbsTol** визначає допустиму абсолютну похібку обчислення (за умовчанням величина параметру дорівнює **1e-6)**; існує можливість задати вектор-рядок абсолютних похибок для кожної компоненти шуканого розв’язку.

Невідомий параметр (вектор невідомих параметрів) вводиться за допомогою команди **bvpinit**, повна форма звернення до якої: **solinit=bvpinit(xinit,yinit,parameters),** де:

- **parameters** – гіпотетичне значення невідомого параметра або вектор гіпотетичних значень невідомих параметрів;
- **solinit** – структура шуканих значень, в якій окрім компонент **solinit.x=xinit** і **solinit.y=y(i:)** використовується ще один компонент: **solinit.parameters=parameters**.

Зауважимо, що ім'я невідомого параметра при зверненні до команди **bvpinit** вибирається довільним чином (наприклад, **bvpinit(x0,y0,lambda)**), навпаки, ім'я компоненти **solinit.parameters** завжди визначено.

Приклад 1. Розв’язати крайову задачу для диференціального рівняння вільних коливань $y''(t)+y(t)=0$ з граничними умовами $y(0)=0,\ y(\pi)=1$ та додатковими параметрами: величина відносної похибки **RelTol=1e-4**, величина абсолютної похибки **AbsTol=1e-7**.

Вихідне диференціальне рівняння замінюємо системою диференціальних рівнянь:

$$\begin{cases} y_1' = y_2 \\ y_2'' = -y_1 \end{cases}.$$

Завдяки тому, що відшукується наближений розв'язок задачі, можна задати додаткові параметри за допомогою оператора **OPTIONS** і команди **bvpset** – відносну та абсолютну похібки:

```
>> OPTIONS=bvpset('AbsTol',1e–7,'RelTol',1e–4)
OPTIONS =
        AbsTol: 1.0000e-007
        RelTol: 1.0000e-004
  SingularTerm: []
     FJacobian: []
    BCJacobian: []
         Stats: []
          Nmax: []
    Vectorized: []
```

Використання команди **linspace** дозволяє задати початкові значення (координати) змінної x у вигляді п'яти рівновіддалених точок на інтервалі $[0\ \pi]$:

```
>> xinit=linspace(0,pi,5);
```

Початкові значення змінної y задаємо у векторній формі:

```
>> yinit=[1 1];
```

За допомогою команди **bvpinit** сформуємо структуру **solinit**, яка використовується для завдання початкових умов для подальшого розв'язання задачі:

```
>> solinit=bvpinit(xinit,yinit)
solinit =
    x: [0  0.7854  1.5708  2.3562  3.1416]
    y: [2x5 double]
```

Безпосередньо розв'язання крайової задачі відбувається за допомогою вже сформованих **m-**файлів **kr1** і **bord**:

```
>> sol=bvp4c(@kr1,@bord,solinit, OPTIONS )
sol =
         x: [1x25 double]
         y: [2x25 double]
        yp: [2x25 double]
    solver: 'bvp4c'
```

Для отримання результату у графічному вигляді треба використати послідовність команд:

```
>> xx=linspace(0,pi/2);yy=deval(sol,xx);plot(xx,yy(1,:)
```

Вправа 3. Розв’язання крайової задачі для звичайних диференціальних рівнянь за допомогою команд підсистеми **Simbolic Math (**аналітичний підхід**)** та чисельними методами у системі **MATLAB**.

Розв’язок крайової задачі здійснюється на основі використання аналітичного методу у разі використання команд підсистеми **Simbolic Math.**

Приклад. Диференціальне рівняння, що описує процес прогибу простої балки постійного перетину з рівномірно розподіленим навантаженням, має вигляд:

$$\frac{d^2u}{dx^2} = \frac{1}{EI}\left(\frac{ql}{2}x - \frac{q}{2}x^2\right),$$

де u – прогин балки в перетині з абсцисою x, EI – жорсткість на вигин перетину балки, q – інтенсивність навантаження, l – довжина балки.

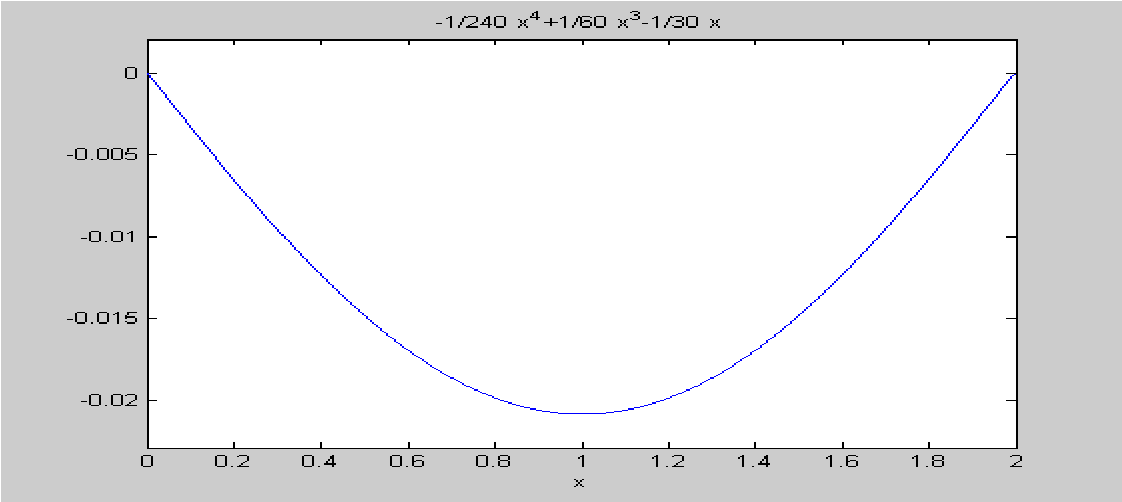
Треба розв’язати крайову задачу з граничними умовами $u(0)=0, u(2)=0$ та заданими значеннями параметрів: $EI=10$, $q=1$, $l=2$.

Розв’язання цієї задачи проводиться за допомогою вже відомої команди **dsolve** [20]. Але на відміну від задачі Коші у вихідних параметрах вказується лише граничні значення шуканої змінної. Для зручності запису в системі **MATLAB** перепозначимо змінну y: $u=y$. Розв’язування диференціального рівняння здійснюємо наступною процедурою:

```
>> y=dsolve('D2y=0.1*x–0.05*x^2','y(0)=0','y(2)=0','x')
y =
–1/240*x^4+1/60*x^3–1/30*x
```

З метою проведення аналізу отриманого результату побудуємо інтегральну криву за домогою команди **ezplot** (див. мал. 15.2):

```
>> ezplot(y,[0 2])
```



Мал. 15.2 – Графік розв’язку диференціального рівняння з граничними умовами за допомогою підсистеми Simbolic Math системи MATLAB

Для порівняння виконаємо розв’язування вихідної крайової задачі за допомогою чисельного алгоритму. Спочатку сформуємо **m**-файли з метою опису вихідного диференціального рівняння та граничних умов:

```
function dydx=kr3(x,y)
dydx=[y(2); 0.1*x-0.05*(x^2)];
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
function res=bord3(ya,yb)
res=[ya(1); yb(1)];
```

Завдання початкових значень здійснюється у формі:

```
>> xinit=linspace(0,2,5)
xinit =
      0    0.5000    1.0000    1.5000    2.0000
>> yinit=[1 1];
>> solinit=bvpinit(xinit,yinit)
solinit =
   x: [0 0.5000 1 1.5000 2]
   y: [2x5 double]
```

Для розв’язання крайової завдачі використовується вже сформовані раніше **m**-файли **kr3** і **bord3:**

```
sol=bvp4c(@kr2,@bord2,solinit);
```

Отримати результат у графічному вигляді можна за допомогою послідовності команд:

```
>> xx=linspace(0,2);
>> yy=deval(sol,xx);
>> plot(xx,yy(1,:)
```

Радимо проаналізувати отримані аналітичним і чисельним методами графікі розв’язку крайової задачі і переконатись в тому, що вони не відрізняються.

Практичні завдання лабороторної роботи № 16

Виконати наступне завдання :

Розв'язати крайову задачу для звичайного диференціального рівняння (варіанти завдань в табл. 16.1) на основі використання:

- аналітичного методу;
- чисельного методу.

Побудувати графіки результатів аналітичного та чисельного розв'язків.

Таблиця 16.1 – Варіанти до завдання № 1

№ п/п	Диференціальне рівняння	Крайові умови	Диференціальне рівняння	Крайові умови
1	$y''(x) = y'(x) - y(x) + 3e^{2x} - 2\sin x$	$y(0) = 5,$ $y(2) = -10$	$y''(x) = -4y(x)$	$y(0) = 5,$ $y(1) = -5$
2	$y''(t) = 10^{-6}\, y'(t) + 10^{-7}(t^2 - 50t)$	$y(0) = 0,$ $y(50) = 0$	$y''(x) = \frac{1}{x}$	$y(0) = 1,$ $y(1) = 1$
3	$y''(x) = y'(x) + y(x) + e^x(1 - 2x)$	$y(0) = 1,$ $y(1) = 3e$	$y''(r) + \frac{1}{r}y'(r) = 0$	$y(1) = \ln 1,$ $y(2) = \ln 2$
4	$y''(t) = -5y'(t) - 6y(t) + te^{-2t} + 3.9\cos 3t$	$y(0) = 0.95,$ $y(3) = 0.15$	$y'' + y^2 = 0$	$y(0) = 1,$ $y(2) = -1$
5	$y''(t) - \frac{1}{t}y'(t) + \frac{1}{t^2}y(t) = 1$	$y(0.5) = 1,$ $y(4.5) = 2$	$y'' = 2y' - y$	$y(0) = 5,$ $y(1) = 10$
6	$y''(t) - \frac{1}{t}y'(t) + \frac{2}{y}(y(t))^2 = 0$	$y(1) = 4,$ $y(2) = 8$	$y'' = e^{2x}$	$y(0) = 0,$ $y(1) = \frac{1}{2}$
7	$y''(t) = \frac{2t}{1+t^2}y'(t) - \frac{2}{1+t^2}y(t) + 1$	$y(0) = 1.25,$ $y(4) = -0.95$	$y'' - \frac{2}{y'+1} = 0$	$y(1) = \frac{1}{3},$ $y(4) = \frac{20}{3}$
8	$y''(t) = \left(-\frac{2}{t}\right)y'(t) + \frac{2}{t^2}y(t) + \frac{10\cos(\ln t)}{t^2}$	$y(1) = 1,$ $y(3) = -1$	$y''(t) = t(y')^2$	$y(0) = \frac{\pi}{2},$ $y(2) = \frac{\pi}{4}$
9	$y''(t) = -2y'(t) - 2y(t) - e^{-t} + \sin 2t$	$y(0) = 0.6,$ $y(4) = -0.1$	$y'' + \frac{1}{y^2}y' = 0$	$y(2) = 2,$ $y(8) = \frac{1}{4}$

Закінчення таблиці 16.1

10	$y''(t) = -4y'(t) - 4y(t) + 5\cos 4 + \sin(2t)$	$y(0) = 0.75$, $y(2) = 0.25$	$y'' - e^y = 0$	$y(0) = 0$, $y(1) = 0$
11	$y''(t) + \frac{1}{t} y'(t) + \left(1 - \frac{1}{t^2}\right) y = 0$	$y(1) = 1$, $y(6) = 0$	$y'' = \sin x$	$y(0) = 0$, $y(2) = 0$
12	$y''(t) = -2y(t) + \sin t$	$y(0) = 0$, $y(1) = 0$	$y'' = x - e^x$	$y(0) = 0$, $y(1) = 0$
13	$y''(t) + \frac{2}{t} y'(t) - \frac{2}{t^2} y(t) = \frac{\sin t}{t^2}$	$y(1) = -0.02$, $y(6) = 0.02$	$(y'')^2 + x^2 = 1$	$y(0) = 1$, $y(1) = 0$
14	$y''(t) + \frac{1}{t} y'(t) + \left(1 - \frac{1}{4t^2}\right) y(t) = \sqrt{t} \cos t$	$y(1) = 1$, $y(6) = -0.5$	$y'' = 5y'$	$y(0) = 0$, $y(1) = 1$

Контрольні питання

1. Визначити поняття крайової задачі і процедури її розв’язання.
2. За допомогою яких команд у системі **MATLAB** можна розв’язувати крайову задачу?
3. В якій формі задаються початкові умови при розв'язуванні крайової задачі в системі **MATLAB**?
4. Пояснити синтаксис команди: **sol=bvp4c(odefun,bcfun,solinit).**
5. Навести етапи розв’язання крайової задачі у системі **MATLAB**.
6. При розв’язанні крайової задачі які додаткові параметри треба визначати для управління якістю результату?
7. Навести етапи розв’язання крайової задачі за допомогою команд підсистеми **Simbolic Math.**