Модуль 6. Жай дифференциалдық теңдеулерді сандық шешу.

6.2 Шеттік есепті шешудің сандық әдістері

$$y'' = f(x, y, y'), \qquad y(a) = \alpha, \qquad y(b) = \beta.$$

дифференциалдық теңдеуін шешіңіз.

Кіріспе

Екі нүктелі шекаралық есептерде дифференциалдық теңдеумен байланысты шекаралық шарттар деп аталатын, көмекші шарттар x-тің екі түрлі мәнінде көрсетіледі. Бұл бастапқы мән есебінен шамалы өзгешелеу ауытқуының үлкен салдары бар — бұл шекаралық мән есептерін шешуді айтарлықтай қиындатады. Бастапқы мән есебінде біз бастапқы мәндер берілген нүктеден басталып, шешімді қажетінше алға жылжыта алдық. Бұл әдіс шекаралық есептер үшін жұмыс істемейді, себебі екі соңғы нүктеде бірегей шешімді шығару үшін қол жетімді бастапқы шарттар жеткіліксіз.

Бастапқы шарттардың жоқтығын жеңудің бір жолы – жетіспейтін бастапқы мәнді болжау. Алынған шешімнің екінші жағындағы шекаралық шарттарды қанағаттандыру ықтималдығы өте аз, бірақ сәйкессіздікті тексеру арқылы біз қайтадан интегралдау алдында бастапқы шарттарға қандай өзгерістер енгізу керектігін бағалай аламыз. Бұл қайталанатын процедура ату әдісі ретінде белгілі. Бұл атау нысанаға ату аналогиясынан алынған — оқ атып, оның нысананың қай жеріне тиетінін бақылаңыз; содан кейін көздеуді түзетіп, қайтадан атыңыз.

Екі нүктелі шекаралық есептерді шешудің тағы бір құралы болып дифференциалдық теңдеулер біркелкі орналасқан тор нүктелерінде ақырлы айырымдармен жуықталатын ақырлы айырымдар әдісі болып табылады. Нәтижесінде дифференциалдық теңдеу алгебралық теңдеулер жүйесіне түрленеді.

Екі әдістің ортақ мәселесі бар: егер дифференциалдық теңдеу сызықты болмаса, олар сызықтық емес теңдеулер жүйесіне алып келеді. Төртінші модульде атап өткеніміздей, сызықтық емес теңдеулерді шешудің барлық әдістері көптеген есептеу ресурстарын тұтынатын итерациялық болып табылады. Осылайша, сызықтық емес шекаралық есептерді шешу жылдам емес. Тағы бір күрделілік - итерациялық әдістерге жақындау үшін жеткілікті жақсы бастапқы мәндер қажет. Бұл бастапқы мәндерді анықтау үшін белгіленген формула болмағандықтан, сызықтық емес шекаралық есептерді шешу алгоритмі бастапқы енгізуді қажет етеді; оны кез келген етіп, ойдан ала салуға болмайды.

Ату әдісі

Ең қарапайым екі нүктелі шекаралық есеп – екінші ретті дифференциалдық теңдеу бір шарты x=a кезінде, екіншісі x=b кезінде көрсетілген. Міне, осындай есептің мысалы:

$$y'' = f(x, y, y'), \qquad y(a) = \alpha, \qquad y(b) = \beta \qquad (1)$$

Енді (1)есебін бастапқы мән есебіне айналдыруға тырысайық

$$y'' = f(x, y, y'), \qquad y(a) = \alpha, \qquad y'(a) = u \qquad (2)$$

Мақсатымыз – u-дың дұрыс мәнін табу. Мұны сынақ және қателік арқылы жасауға болады: u деп болжап және $x = a$-дан b-ға дейін жүру арқылы бастапқы мән есебін шешу. Егер шешім белгіленген $y(b) = \beta$ шекаралық шартымен сәйкес келсе, біз есепті шештік; әйтпесе u реттеп, әрекетті қайталауымыз керек. Бұл процедура өте жалықтыратыны анық.

Егер біз u анықтаудың түбір табу мәселесі екенін түсінсек, бізге жүйелі бірнеше әдістер қолжетімді болады. Бастапқы мән есебінің шешімі u-ға тәуелді болғандықтан, $y(b)$-тің есептелген мәні u-дан тәуелді функциясы; яғни

$$y(b) = \theta(u)$$

Демек, u

$$r(u) = \theta(u) - \beta = 0 \qquad (3)$$

мәнінің түбірі, мұндағы $r(u)$ – шекаралық қалдық ($x = b$ кезінде есептелген және көрсетілген шекаралық мән арасындағы айырмашылық). (3) теңдеуді 4-модульде қарастырылған түбірлерді табу әдістерінің бірімен шешуге болады. Біз екіге бөлу әдісін қабылдамаймыз, себебі ол $\theta(u)$ тым көп бағалауды қамтиды. Ньютон-Рафсон әдісінде біз $d\theta/du$ есептеу мәселесіне тап боламыз, оны жасауға болады, бірақ оңай емес. Бұдан Риддер алгоритмін таңдауы қалады.

Сызықты емес шекаралық есептерді шешуде қолданатын процедурамыз:

- (3) теңдеуінің u түбірі үшін u_1 және u_2 бастапқы мәндер аралығын таңдап алу керек.
- (3) теңдеуді шешу үшін Риддер әдісін қолданыңыз. Әрбір итерация дифференциалдық теңдеуді бастапқы мән есебі ретінде шешу арқылы $\theta(u)$ мәнін бағалауды қажет ететінін ескеріңіз.
- u мәнін анықтап, дифференциалдық теңдеулерді тағы бір рет шешіп, нәтижелерін жазып алыңыз.

Егер дифференциалдық теңдеу сызықтық болса, кез келген түбір табу әдісі u анықтау үшін тек бір ғана интерполяция қажет болады. Риддер әдісі үш нүктені (u_1, u_2 және u_3) пайдаланатындықтан, ол тек екі нүктені ($u_1,$ және u_2) пайдаланатын сызықтық интерполяциямен салыстырғанда ысырап болып табылады. Сондықтан дифференциалдық теңдеу сызықтық болған кезде Риддер әдісін сызықтық интерполяцияға ауыстырамыз.

МЫСАЛ 1

$$y'' + 3yy' = 0 \qquad y(0) = 0 \qquad y(2) = 1$$

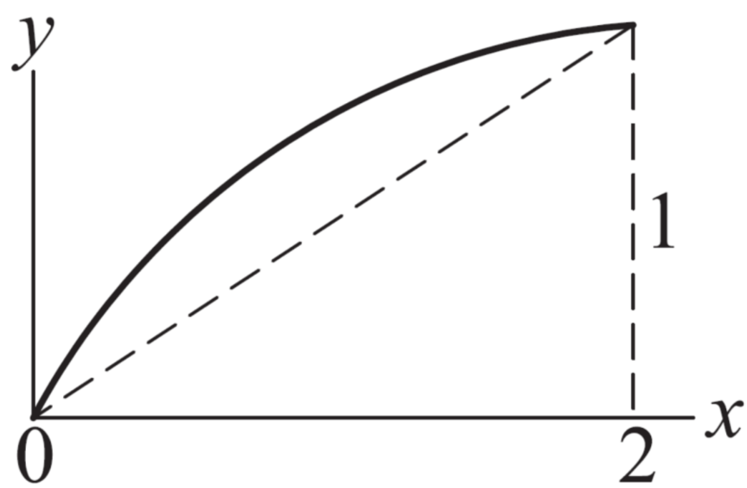
шектік есепті шешіңіз.

Шешуі. Эквивалентті бірінші ретті теңдеулер

$$y' = \begin{bmatrix} y_0' \\ y_1' \end{bmatrix} = \begin{bmatrix} y_1 \\ -3y_0y_1 \end{bmatrix}$$

шекаралық шарттарымен $y_0(0) = 0y_0(2) = 1$ болады.

Енді $y'(0)$ сынақ мәндерін анықтау қиын есебі келеді. Біз әрқашан кездейсоқ екі санды таңдап, ең жақсысына үміттене аламыз. Дегенмен, аздаған іздеу жұмысы арқылы кездейсоқтық элементін азайтуға болады. Біз $0 \leq x \leq 2$ интервалында y тегіс (тербелмейді) деген негізді болжам жасаудан бастаймыз. Әрі қарай y 0-ден 1-ге дейін ұлғаюы керек екенін ескереміз, бұл үшін $y' > 0$ қажет. Өйткені y да, y' да оң болса, дифференциалдық теңдеуді қанағаттандыру үшін y'' теріс болуы керек деген қорытындыға келеміз. Енді біз y-тің дөрекі сызбасын жасай аламыз:



Кескінге қарап, $y'(0) > 0{,}5$ екені анық, осылайша $y'(0)$ жақшалары үшін $y'(0) = 1$ және 2 орынды мәндер болып көрінеді; егер олар болмаса, Риддер әдісі қате туралы хабарды көрсетеді.

Келесі тізімде келтірілген бағдарламада интегралдау үшін төртінші ретті Рунге-Кутта әдісін таңдалды. Оны импорттау мәлімдемесіндегі run_kut4 орнына run_kut5 арқылы адаптивті нұсқамен ауыстыруға болады. Есепті сипаттау үшін пайдаланушы қамтамасыз ететін үш функция қажет екенін ескеріңіз. Дифференциалдық теңдеулерді анықтайтын F(x,y) функциясынан басқа интегралдаудың бастапқы шарттарын анықтау үшін initCond(u) функциялары, ал Риддер әдісін қалдық шекаралық шартпен қамтамасыз ету үшін r(u) функциялары қажет. Осы функциялардағы бірнеше операторларды өзгерту арқылы бағдарламаны кез келген екінші ретті шекаралық есептерге қолдануға болады. Ол сондай-ақ үшінші ретті теңдеу үшін де орындалады, егер интегралдаудың үш шекаралық шарттың екеуі соңында көрсетілгеннен басталса.

```
## example1
import numpy as np
from run_kut4 import *
from ridder import *
from printSoln import *
def initCond(u): # [y,y’] бастапқы мәндер; ’u’белгісіз
    return np.array([0.0, u])

def r(u):  # Шекаралық шарт қалдығы – (8.3)теңдеуді қараңыз.
    X,Y = integrate(F,xStart,initCond(u),xStop,h)
    y = Y[len(Y) - 1]
    r = y[0] - 1.0
    return r
def F(x,y): # Бірінші ретті дифференциалдық теңдеулер
    F = np.zeros(2)
    F[0] = y[1]
    F[1] = -3.0*y[0]*y[1]
    return F

xStart = 0.0          # Start of integration
xStop = 2.0           # End of integration
u1 = 1.0              # 1st trial value of unknown init. cond.
u2 = 2.0              # 2nd trial value of unknown init. cond.
h = 0.1               # Step size
```

```
freq = 2              # Printout frequency
u = ridder(r,u1,u2)   # Compute the correct initial condition
X,Y = integrate(F,xStart,initCond(u),xStop,h)
printSoln(X,Y,freq)

Шешімі:
    x         y[ 0 ]       y[ 1 ]
0.0000e+00 0.0000e+00 1.5145e+00
2.0000e-01 2.9404e-01 1.3848e+00
4.0000e-01 5.4170e-01 1.0743e+00
6.0000e-01 7.2187e-01 7.3287e-01
8.0000e-01 8.3944e-01 4.5752e-01
1.0000e+00 9.1082e-01 2.7013e-01
1.2000e+00 9.5227e-01 1.5429e-01
1.4000e+00 9.7572e-01 8.6471e-02
1.6000e+00 9.8880e-01 4.7948e-02
1.8000e+00 9.9602e-01 2.6430e-02
2.0000e+00 1.0000e+00 1.4522e-02
```

$y'(0) = 1.5145$ екенін ескеріңіз, сондықтан 1,0 және 2,0 бастапқы мәндері дұрыс таңдалды.

Қолданылған әдебиеттер тізімі

1) Шакенов Қ.Қ. Есептеу математикасы әдістері лекциялар курсы.Алматы,2019. – 193б
2) P. Dechaumphai, N. Wansophark. Numerical Methods in Science and Engineering Theories with MATLAB, Mathematica, Fortran, C and Python Programs. Alpha Science International Ltd. 2022
3) Н. С. Бахвалов, Н. П. Жидков, Г. М. Кобельков Численные методы: Классический университетский учебник. —М.: Издательство «Бином. Лаб. знаний»,2020. — 636 с.
4) Вабищевич П.Н. Численные методы: Вычислительный практикум. — М.: Книжный дом «ЛИБРОКОМ», 2020. — 320 с.

Интернет ресурстар

1) https://docs.python.org/3/
2) http://math-hse.info/f/2018-19/py-polit/instruction_JN.pdf
3) https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/execute.html
4) https://colab.research.google.com/
5) https://planetcalc.ru/search/?tag=2874