Цена 20 коп.

АКАДЕМИЯ НАУК СССР
ВЫЧИСЛИТЕЛЬНЫЙ ЦЕНТР

СООБЩЕНИЯ ПО ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ ЭВМ

ПАКЕТ ПРИКЛАДНЫХ ПРОГРАММ ДЛЯ РЕШЕНИЯ ЛИНЕЙНЫХ ДВУХТОЧЕЧНЫХ КРАЕВЫХ ЗАДАЧ

ВЫЧИСЛИТЕЛЬНЫЙ ЦЕНТР АН СССР
МОСКВА 1982

АКАДЕМИЯ НАУК СССР
ВЫЧИСЛИТЕЛЬНЫЙ ЦЕНТР

СООБЩЕНИЯ ПО ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ ЭВМ

А.А.АБРАМОВ, Н.Г.БУРАГО, В.В.ДИТКИН, А.Л.ДЫШКО, А.Ф.ЗАБОЛОЦКАЯ,
Н.Б.КОНЮХОВА, Б.С.ПАРИЙСКИЙ, В.И.УЛЬЯНОВА, И.И.ЧЕЧЕЛЬ

ПАКЕТ ПРИКЛАДНЫХ ПРОГРАММ ДЛЯ РЕШЕНИЯ ЛИНЕЙНЫХ ДВУХТОЧЕЧНЫХ КРАЕВЫХ ЗАДАЧ

ВЫЧИСЛИТЕЛЬНЫЙ ЦЕНТР АН СССР
МОСКВА 1982

УДК 519.624

Ответственный редактор
доктор физ.-матем. наук В.А.Диткин

В работе дается краткое описание методов и алгоритмов, реализованных в пакете прикладных программ LTPBVP. Пакет предназначен для решения двухточечных краевых задач для линейных обыкновенных дифференциальных и разностных уравнений, а также систем таких уравнений. Пакет содержит в виде отдельных процедур и подпрограмм различные варианты дифференциальной и разностной прогонки. Процедуры и подпрограммы написаны на АЛГОЛе или ФОРТРАНе.

Введение

В Вычислительном центре АН СССР создан пакет прикладных программ "Линейные обыкновенные краевые задачи", предназначенный для решения двухточечных краевых задач для линейных обыкновенных дифференциальных и разностных уравнений, а также систем таких уравнений. Пакет является частью системы пакетов прикладных программ для реализации общих методов вычислительной математики и включен в вычислительный комплекс САФРА (см. [1]). Пакет состоит из двух частей: систематизированной библиотеки процедур и программ с именем LTPBVP (Linear Two Point Boundary Value Problem) и пакета SPARSE. Здесь речь пойдет о пакете LTPBVP, о пакете SPARSE см. [2-4]. Часть программ пакета LTPBVP написана на языке ФОРТРАН, часть на языке АЛГОЛ. Реализация пакета осуществлялась на ЭВМ БЭСМ-6.

В пакет LTPBVP включены как методы решения краевых дифференциальных задач, т.е. задач, сформулированных в виде системы линейных обыкновенных дифференциальных уравнений с коэффициентами, заданными какими-либо формулами, и соответствующих граничных условий, так и методы решения разностных краевых задач, сформулированных в виде системы линейных алгебраических уравнений специального типа. В пакете LTPBVP реализованы методы решения двухточечных краевых задач, задач с условием периодичности и общей двухточечной задачи с нераспадающимися краевыми условиями. Пакет содержит в виде отдельных процедур и подпрограмм различные варианты дифференциальной и разностной прогонки.[1] Процедуры пакета LTPBVP сданы в Государственный фонд алгоритмов и программ при ВЦ АН СССР (тексты процедур, комментарии к этим текстам и тестовые примеры). В [25-41] приведены краткие сведения об этих материалах.

Часть I. Дифференциальная прогонка.

Некоторые обозначения и предложения

о.д.у. – обыкновенные дифференциальные уравнения.

---

[1] В пакет не включены методы, предложенные П.И.Монастырным (см., например, [42-46]), так как авторы не имели опыта работы с этими методами.

с.о.д.у. – системы о.д.у.

x – независимое переменное, всегда вещественное.

$[a, b]$ – отрезок интегрирования, может быть $a \geq b$ или $a \leq b$

$y, y_1, \ldots, y_n$ – искомые функции (или вектор-функции), вещественные или комплексные в соответствии с тем, как это оговорено в рассматриваемых случаях.

n – порядок рассматриваемой с.о.д.у., задаваемое положительное число.

m – задаваемое положительное число. Ответ для y будет вычислен в $m+1$ точке.

$x_0, x_1, \ldots, x_m$ – те значения x, для которых должен быть вычислен ответ.

$p, q, f, P, P_{11}, P_{12}, P_{21}, P_{22}, f_1, f_2$ – коэффициенты и правые части уравнений, вещественные или комплексные в соответствии с тем, как это оговорено в рассматриваемых случаях, заданные на $[a, b]$ функции (или матричные функции). Предлагаемые процедуры производят вычисление при любых коэффициентах и правых частях. Однако следует учесть, что для решения возникающих вспомогательных о.д.у. используется метод Рунге-Кутта, поэтому негладкость этих заданных функций может привести к практической нереализуемости процедуры.

Всюду индексом a внизу отмечается решение задачи Коши от a к b, индексом b внизу – решение задачи Коши от b к a.

В качестве нормы матрицы C, $C=(c_{ij})$ (соответственно нормы вектора d, $d=(d_j)$) используется следующая: $\|C\| = \max_{i,j} |c_{ij}|$ (соответственно $\|d\| = \max_j |d_j|$).

Все вспомогательные процедуры, используемые для решения возникающих вспомогательных задач (задача Коши для с.о.д.у., задача решения систем линейных алгебраических уравнений и др.), включены в тела предлагаемых процедур. Привлечение каких-либо библиотечных процедур не требуется. Все получаемые величины хранятся во внутренней памяти.

Возможно, что для некоторых классов задач методы, выбранные для реализации вспомогательных процедур, не являются удачными. В частности, для решения задач Коши используется метод Рунге-Кутта. Тем самым, если возникающие с.о.д.у. являются жесткими, то рекомендуемые процедуры повлекут очень существенное измельчение шага интегрирования и, возможно, станут неэффективными. В подобных случаях можно рекомендовать переделку предлагаемых процедур (например, замену метода Рунге-Кутта на метод Гира).

Описание методов приведено в том виде, как они реализованы процедурами пакета LTPBVP.

Идентификаторы процедур, содержащихся в первой части, составлены по следующему правилу. Последний символ – буква d (дифференциальная прогонка), перед этим $s1$ (для систем уравнений первого порядка) или $s2$ (для систем уравнений второго порядка), или o (для одного уравнения второго порядка); перед этим несколько букв, напоминающих название метода. Примеры:

clod – classical differential method for one equation of the second order;

ms1d – Moszyński differential method for the first (1) order systems of differential equations.

Глава I. Одно дифференциальное уравнение второго порядка

Для решения двухточечных краевых задач для одного о.д.у. второго порядка в пакете прикладных программ LTPBVP реализованы следующие методы дифференциальной прогонки: классический и потоковый варианты метода прогонки для решения самосопряженной задачи с разделенными краевыми условиями; циклическая прогонка для решения самосопряженной задачи с условиями периодичности решения; два варианта ортогональной прогонки для решения общей краевой задачи с вещественными или комплексными коэффициентами и разделенными краевыми условиями.

Описание и исследование этих методов см. в [5-12]; сообщения об их реализации в пакете LTPBVP в виде стандартных процедур на языке АЛГОЛ см. в [25-27].

§ I. Формулировка задач

I.I. Самосопряженные задачи

Самосопряженная двухточечная краевая задача с разделенными краевыми условиями для одного о.д.у. второго порядка имеет вид

$$(p(x)y')' - q(x)y = f(x), \quad x \in [a, b], \qquad (I.1)$$

$$\alpha_1 y'(a) - \beta_1 y(a) = \gamma_1 , \qquad (I.2)$$

$$\alpha_2 y'(b) + \beta_2 y(b) = \gamma_2 . \qquad (I.3)$$

Здесь $p(x)$, $p'(x)$, $q(x)$, $f(x)$ – непрерывные на $[a,b]$ функции, принимающие вещественные значения, α_j, β_j, γ_j – заданные числа, $\alpha_j^2 + \beta_j^2 > 0$, $j = 1,2$.

Известно, что по крайней мере при дополнительных предположениях

$$p(x) \geq p_0 > 0, \quad q(x) \geq q_0 > 0 \quad \text{на } [a,b],$$

$$\alpha_j \geq 0, \quad \beta_j \geq 0, \quad \alpha_j + \beta_j > 0, \quad j = 1,2, \qquad (I.4)$$

задача (I.I) – (I.3) имеет единственное решение; это решение обладает двумя непрерывными производными (см. [7]). Те же условия (I.4) являются достаточными для устойчивости метода классической прогонки [5–8] применительно к решению задачи (I.I) – (I.3). В пакете *LTPBV* метод классической прогонки для решения задачи (I.I) – (I.3) реализован процедурой *clod* [25].

Кроме уравнения (I.I) с непрерывными на $[a,b]$ коэффициентами, на практике часто возникает уравнение вида

$$\left(\frac{y'}{\sigma(x)}\right)' - q(x)y = f(x), \qquad x \in [a,b], \qquad (I.5)$$

где $\sigma(x)$ может обращаться в нуль во внутренних точках отрезка $[a,b]$. Например, для уравнения теплопроводности может оказаться, что на некоторых интервалах внутри рассматриваемой области имеются изотермические участки, где коэффициент теплопроводности k, $k = 1/\sigma$, обращается в бесконечность – и в уравнении (I.5) появляется особенность. При этом функция ψ, $\psi = y'/\sigma$, остается конечной и непрерывной во всей области и ее вычисление представляет самостоятельный интерес. Для решения задачи (I.5), (I.2), (I.3) в [9] предложен вариант "потоковой" прогонки – модифицированный метод классической прогонки, в котором за зависимую переменную принимается функция ψ, называемая "потоком" (отсюда и название метода). Метод [9] не приводит к особенностям при обращении $\sigma(x)$ в нуль во внутренних точках отрезка $[a,b]$. В пакете *LTPBVP* метод потоковой прогонки для решения задачи (I.5), (I.2), (I.3) реализован процедурой *stclod* [26].

К распространенным задачам для уравнения (I.I) относится задача отыскания решения, удовлетворяющего граничным условиям периодичности:

$$y(a) = y(b), \quad y'(a) = y'(b) \qquad (I.6)$$

Для решения задачи (I.I), (I.6) в [I0] предложен вариант "циклической" прогонки, основанный на решении двух вспомогательных неоднородных краевых задач с распавшимися краевыми условиями. В пакете *LTPBVP* метод циклической прогонки для решения задачи (I.I), (I.6) реализован процедурой *cyclod* [25]. В этой процедуре для решения вспомогательных линейных неоднородных краевых задач используется метод классической прогонки.

I.2. Общие краевые задачи

I.2.I. Общая двухточечная краевая задача с разделенными краевыми условиями для одного о.д.у. второго порядка имеет вид

$$y'' + p(x)y' + q(x)y = f(x), \quad x \in [a,b], \qquad (I.7)$$

$$\alpha_1 y(a) + \beta_1 y'(a) = \gamma_1 , \qquad (I.8)$$

$$\alpha_2 y(b) + \beta_2 y'(b) = \gamma_2 . \qquad (I.9)$$

Здесь $p(x)$, $q(x)$, $f(x)$ – непрерывные на $[a,b]$ функции, принимающие вещественные или комплексные значения; α_j, β_j, γ_j – вещественные или комплексные числа; $|\alpha_j|^2 + |\beta_j|^2 > 0$, $j = 1,2$.

Для решения задачи (I.7) – (I.9) с вещественными коэффициентами в [II] предложен вариант ортогональной прогонки, основанный на идее ортогонального преобразования переменных. Показано, что этот метод устойчив, если исходная краевая задача (I.7) – (I.9) устойчива относительно малых изменений определяющих ее величин. В пакете *LTPBVP* метод ортогональной прогонки [II] для решения задачи (I.7) – (I.9) с вещественными коэффициентами реализован процедурой *aod* [27].

Для решения задачи (I.7) – (I.9) с комплексными коэффициентами используется вариант ортогональной прогонки, основанный на идее ортогонального переноса граничных условий и предложенный в [I2] для случая однородной задачи. Этот метод устойчив настолько, насколько хорошо обусловлена исходная краевая задача. В пакете *LTPBVP* метод ортогональной прогонки [I2] для решения задачи (I.7) – (I.9) с комплексными коэффициентами реализован процедурой *bicod* [27].

I.2.2. Общая двухточечная задача с нераспавшимися краевыми условиями имеет вид уравнения (I.7) и граничных условий

$$\alpha_1 y(a) + \alpha_2 y'(a) + \alpha_3 y(b) + \alpha_4 y'(b) = \gamma_1 , \qquad (I.I0)$$

$$\beta_1 y(a) + \beta_2 y'(a) + \beta_3 y(b) + \beta_4 y'(b) = \gamma_2 . \qquad (I.II)$$

Используя прием, предложенный в [21], введем новые искомые функции $z_1(x)$, $z_2(x)$: $z_1(x) = y(x)$, $z_2(x) = y(a+b-x)$.

Для $z(x)$, $z(x) = \begin{Vmatrix} z_1(x) \\ z_2(x) \end{Vmatrix}$, из (I.7), (I.I0), (I.II) получим задачу с разделенными краевыми условиями:

$$z'' + P(x)\,z' + Q(x)\,z = F(x), \qquad a \le x \le \frac{a+b}{2}, \tag{I.I2}$$

$$S_1 z(a) + C_1 z'(a) = g_1, \tag{I.I3}$$

$$S_2 z\left(\frac{a+b}{2}\right) + C_2 z'\left(\frac{a+b}{2}\right) = 0, \tag{I.I4}$$

где $P(x) = \begin{Vmatrix} P(x) & 0 \\ 0 & -P(a+b-x) \end{Vmatrix}$,

$Q(x) = \begin{Vmatrix} q(x) & 0 \\ 0 & q(a+b-x) \end{Vmatrix}$, $F(x) = \begin{Vmatrix} f(x) \\ f(a+b-x) \end{Vmatrix}$,

$S_1 = \begin{Vmatrix} \alpha_1 & \alpha_3 \\ \beta_1 & \beta_3 \end{Vmatrix}$, $C_1 = \begin{Vmatrix} \alpha_2 & -\alpha_4 \\ \beta_2 & -\beta_4 \end{Vmatrix}$,

$g_1 = \begin{Vmatrix} \gamma_1 \\ \gamma_2 \end{Vmatrix}$, $S_2 = \begin{Vmatrix} 1 & -1 \\ 0 & 0 \end{Vmatrix}$, $C_2 = \begin{Vmatrix} 0 & 0 \\ 1 & 1 \end{Vmatrix}$.

Задача (I.I2) - (I.I4) может быть решена одним из методов главы П или главы Ш, предназначенных для решения систем уравнений.

Замечание.

При реализации методов дифференциальной прогонки [5-I2] для решения описанных двухточечных краевых задач возникает следующая трудность. Для осуществления обратной прогонки необходимо хранить результаты, полученные при прямой прогонке, во всех точках интегрирования (или, во всяком случае, в достаточно часто расположенных узлах сетки). Чтобы избежать этого, в алгоритмах [25-27] методы [5-I2] реализованы в так называемых "двухсторонних" вариантах, что позволяет хранить промежуточную информацию только в тех точках, где нас интересует решение исходной задачи.

§ 2. Классический и потоковый варианты метода прогонки для решения самосопряженных задач

Метод классической прогонки [5-8] для решения задачи (I.I) - (I.3) в пакете LTPBVP реализован в двухстороннем варианте процедурой clod [25]; метод потоковой прогонки [9] для решения задачи (I.5), (I.2), (I.3) - процедурой stclod [26].

2.I. Описание методов

Ниже дано описание алгоритма, реализуемого процедурой clod для решения задачи (I.I) - (I.3). Те же формулы с заменой $\frac{1}{P(x)} = \sigma(x)$ соответствуют алгоритму, реализуемому процедурой stclod для решения задачи (I.5), (I.2), (I.3). (В процедуре stclod в отличие от процедуры clod вместо производной $y'(x)$ вычисляется величина потока $\psi(x) = y'(x)/\sigma(x)$; другие отличия алгоритма [26] от приведенного ниже несущественны, и мы их указывать не будем.)

I) Случай I: $|\alpha_1| \ge |\beta_1|$, $|\alpha_2| \ge |\beta_2|$.

По всему отрезку $[a, b]$ переносим соотношение

$$p(x)\,y' = \alpha(x)\,y + \beta(x), \tag{2.I}$$

решая от a к b соответствующие задачи Коши для уравнений классической прогонки:

$$\alpha' + \alpha^2/p(x) - q(x) = 0, \qquad x \in [a, b], \tag{2.2}$$

$$\alpha(a) = \beta_1 p(a)/\alpha_1; \tag{2.3}$$

$$\beta' + \frac{\alpha(x)}{p(x)}\,\beta = f(x), \qquad x \in [a, b], \tag{2.4}$$

$$\beta(a) = p(a)\gamma_1/\alpha_1. \tag{2.5}$$

При этом запоминаем значения $\alpha_a(x_s)$, $\beta_a(x_s)$ в тех точках x_s, $s = 0, 1, \dots, m$, где нас интересует решение исходной задачи. От b к a интегрируем уравнения (2.2), (2.4) с условиями

$$\alpha(b) = -p(b)\,\beta_2/\alpha_2, \qquad \beta(b) = p(b)\,\gamma_2/\alpha_2,$$

вычисляя $\alpha_b(x_s)$, $\beta_b(x_s)$, $s = m, m-1, \dots, 0$.

Решение $y(x)$ задачи (I.I) - (I.3) и его производную $y'(x)$ определяем в точках x_s отрезка $[a, b]$ по формулам

$$y(x_s) = \frac{\beta_b(x_s) - \beta_a(x_s)}{\alpha_b(x_s) - \alpha_a(x_s)},$$

$$y'(x_s) = \frac{\beta_a(x_s)\,\alpha_b(x_s) - \beta_b(x_s)\,\alpha_a(x_s)}{p(x_s)\,[\alpha_b(x_s) - \alpha_a(x_s)]}, \tag{2.6}$$

$$s = m, \quad m-1, \dots, 0.$$

2) Случай П: $|\alpha_1| \ge |\beta_1|$, $|\alpha_2| < |\beta_2|$.

От a к b переносим соотношение (2.I), решая задачи Коши (2.2), (2.3) и (2.4), (2.5). От b к a переносим соотношение

$$\varphi(x)\,p(x)\,y' - y = \beta(x), \tag{2.7}$$

решая соответствующие задачи Коши для уравнений прогонки:

$$\varphi' + q(x)\,\varphi^2 = 1/p(x)\ , \qquad x \in [a,b]\ , \tag{2.8}$$

$$\varphi(b) = 0\ ; \tag{2.9}$$

$$\beta' + q(x)\,\varphi(x)\,\beta = \varphi(x)\,f(x)\ , \quad x \in [a,b]\ , \tag{2.I0}$$

$$\beta(b) = -\gamma_2/\beta_2\ . \tag{2.II}$$

Решение задачи (I.I) - (I.3) определяется по формулам

$$y(x_s) = \frac{\varphi_\beta(x_s)\,\beta_\alpha(x_s) - \beta_\beta(x_s)}{1 - \varphi_\beta(x_s)\,\alpha_\alpha(x_s)}\ ,$$

$$y'(x_s) = \frac{\beta_\alpha(x_s) - \alpha_\alpha(x_s)\,\beta_\beta(x_s)}{p(x_s)\,[1 - \varphi_\beta(x_s)\,\alpha_\alpha(x_s)]}\ , \quad s = m, m-1, \ldots, 0\ . \tag{2.I2}$$

3) Случай Ш: $|\alpha_1| < |\beta_1|$, $|\alpha_2| \geq |\beta_2|$.

Сводится к случаю II, если поменять ролями концы отрезка $[a,b]$

4) Случай IУ: $|\alpha_1| < |\beta_1|$, $|\alpha_2| < |\beta_2|$.

По всему отрезку $[a,b]$ переносим соотношение (2.7), от a к b интегрируя уравнения (2.8), (2.I0) с граничными условиями

$$\varphi(a) = 0\ , \qquad \beta(a) = \gamma_1/\beta_1\ ,$$

а от b к a решая задачи Коши (2.8), (2.9) и (2.I0), (2.II). Решение задачи (I.I) - (I.3) определяем по формулам

$$y(x_s) = \frac{\varphi_\beta(x_s)\,\beta_\alpha(x_s) - \varphi_\alpha(x_s)\,\beta_\beta(x_s)}{\varphi_\alpha(x_s) - \varphi_\beta(x_s)}\ ,$$

$$y'(x_s) = \frac{\beta_\alpha(x_s) - \beta_\beta(x_s)}{p(x_s)\,[\varphi_\alpha(x_s) - \varphi_\beta(x_s)]}\ , \quad s = m, m-1, \ldots, 0\ . \tag{2.I3}$$

2.2. Численная реализация методов

Алгоритмы [25,26] дают решение в точках x_s отрезка $[a,b]$, задаваемых формулой

$$x_s = a + s(b-a)/m\ , \qquad s = 0, 1, \ldots, m\ . \tag{2.I4}$$

Граничное условие в точке $a\ (a = x_0)$ переносится в точку x_1, затем из x_1 в x_2, ... , из x_{m-1} в $x_m\ (x_m = b)$. Аналогично переносится в каждую из точек x_s $(s = m-1, \ldots, 0)$ граничное условие в точке b.

Численное решение вспомогательных задач Коши осуществляется мет

дом Рунге-Кутта четвертого порядка точности с автоматическим выбором шага. Для интегрирования на один шаг используется алгоритм, изложенный в процедуре RK1ST [47]. Контроль точности и выбор шага осуществляются следующим образом. Разобьем отрезок интегрирования $[x_s, x_{s+1}]$ на m_1 частей и положим $h = (x_{s+1} - x_s)/m_1$. Вычислим по формулам Рунге-Кутта $y_h(x_{s1})$, $y_{h/2}(x_{s1})$, $y'_h(x_{s1})$, $y'_{h/2}(x_{s1})$, где $x_{s1} = x_s + h$; $y_h(x_{s1})$, $y'_h(x_{s1})$ - значения функции и производной в точке x_{s1}, вычисленные при шаге h, а $y_{h/2}(x_{s1})$, $y'_{h/2}(x_{s1})$ - значения функции и производной в той же точке x_{s1}, вычисленные при шаге $\hat{h}$, $\hat{h} = h/2$. Шаг интегрирования выбирается таким образом, чтобы в точке x_{s1} выполнялись условия

$$|y_h(x_{s1}) - y_{h/2}(x_{s1})| < \frac{eps}{m \times m1}\ ,$$

$$|y'_h(x_{s1}) - y'_{h/2}(x_{s1})| < \frac{eps}{m \times m1}\ , \tag{2.I5}$$

где eps - заданная точность вычислений. Если условия (2.I5) не выполняются, то шаг интегрирования уменьшается вдвое, т. е. m_1 удваивается. Если условия (2.I5) выполняются, то шаг интегрирования удваивается, что равносильно уменьшению вдвое m_1. Поэтому первоначально m_1 задается в виде $m_1 = 2^k$, $0 \leq k$ - целое число. Интегрирование от точки x_{s1} до точки x_{s2}, $x_{s2} = x_{s1} + h$, производится с тем наибольшим шагом h, при котором выполнены условия (2.I5). Затем проверяется выполнение аналогичных условий в точке x_{s2}, и т.д. Очевидно, что при фиксированном m и фиксированной точности eps время счета существенно зависит от выбора параметра m_1.

2.3. Описание процедур

Имя процедуры и совокупность формальных параметров имеют вид:

clod (a, b, m, m1, ua, ub, y, y1, pqf, eps, lm, lp)

для метода классической прогонки и

stclod (a, b, m, m1, ua, ub, y, y1, pqf, eps, lm, lp)

для метода потоковой прогонки. Здесь a, b - концы отрезка, на котором решается задача (задача (I.I) - (I.3) для процедуры clod и задача (I.5), (I.2), (I.3) для процедуры stclod); m - целое положительное число, определяющее число точек x_s (зада-

ваемых формулой (2.I4)), в которых будут получены ответы;

$m1$ – целое положительное число вида 2^k, $0 \le k$ – целое, употребляемое пользователем для оптимизации хода вычислений (см. п.2.2, где это число обозначается через m_1);

eps – заданная точность вычислений (см. подробнее п.2.2);

ua – массив $ua[1:3]$, задающий граничные условия в точке a ($ua[1]=\alpha_1$, $ua[2]=\beta_1$, $ua[3]=\gamma_1$);

ub – массив $ub[1:3]$, задающий граничные условия в точке b ($ub[1]=\alpha_2$, $ub[2]=\beta_2$, $ub[3]=\gamma_2$);

y – массив $y[0:m]$ – массив ответов ($y[s]$ – значение $y(x_s)$, $s=0,1,\dots,m$);

$y1$ – массив $y1[0:m]$ – массив ответов ($y1[s]$ – значение $y'(x_s)$ для процедуры $clod$ и значение $y'(x_s)/6(x_s)$ для процедуры $stclod$);

pqf – идентификатор процедуры с четырьмя параметрами, которая должна быть описана в программе, использующей процедуру $clod$ или процедуру $stclod$; процедура $pqf(x, p, q, f)$ вычисляет для данного x значения $p=p(x)$, $q=q(x)$, $f=f(x)$ коэффициентов и правой части уравнения (I.I) при использовании процедуры $clod$ и значения $p=6(x)$, $q=q(x)$, $f=f(x)$ коэффициентов и правой части (I.5) при использовании процедуры $stclod$;

lp – метка, на которую происходит выход из процедуры в случае плохой обусловленности исходной задачи (точнее, если знаменатель в формулах (2.6), (2.I2) или (2.I3) становится по абсолютной величине меньше eps);

lm – метка, на которую происходит выход из процедуры, если данный метод не подходит для решения исходной задачи (точнее, если $(b-a)/h$ 10^{10}, где h – значение шага интегрирования вспомогательной задачи Коши, полученное к моменту выхода на метку lm).

§ 3. Циклическая прогонка для решения самосопряженной задачи с граничными условиями периодичности

Метод циклической прогонки [I0] для решения задачи (I.I), (I.6) в пакете $LTPBVP$ реализован процедурой $cyclod$ [25].

3.I. Описание метода

Для решения задачи (I.I), (I.6) вводятся две вспомогательные

краевые задачи с разделенными краевыми условиями:

$$(p(x)u')' - q(x)u = f(x), \quad x \in [a,b], \quad u(a) = u(b) = 0; \tag{3.I}$$

$$(p(x)w')' - q(x)w = 0, \quad x \in [a,b], \quad w(a) = w(b) = 1. \tag{3.2}$$

Решение задачи (I.I), (I.6) ищется в виде

$$y(x) = u(x) + \lambda w(x),$$

где константа λ подлежит определению, а $u(x)$ и $w(x)$ – решения задач (3.I) и (3.2) соответственно.

Условие $y(a) = y(b)$, очевидно, выполняется. Условие $y'(a) = y'(b)$ дает

$$\lambda = \frac{u'(a) - u'(b)}{w'(b) - w'(a)}. \tag{3.3}$$

Значения $u'(a)$, $u'(b)$, $w'(a)$, $w'(b)$ получаются при использовании метода прогонки для решения вспомогательных задач (3.I) и (3.2).

Если $\frac{1}{p(x)} \ge 0$ и $q(x) \ge 0$, то для решения этих задач можно применять классическую прогонку или потоковую (см. § 2 этой главы), при этом получим $w'(a) < 0$, $w'(b) > 0$.

3.2. Описание процедуры

В процедуре $cyclod$ [25] для решения вспомогательных задач (3.I) и (3.2) используется двусторонний вариант метода классической прогонки (см. п.2.I, случай IУ). Численная реализация решения возникающих задач Коши для уравнений классической прогонки та же, что и в методах § 2 (см. п.2.2).

Имя процедуры и совокупность формальных параметров имеют вид:

$$cyclod(a, b, m, m1, y, y1, pqf, eps, lm, lp).$$

Здесь смысл формальных параметров a, b, m, $m1$, y, $y1$, pqf, eps тот же, что и аналогично поименованных параметров процедуры $clod$ (см. п.2.3); параметры, соответствующие заданию граничных условий, отсутствуют; lm – метка, на которую происходит выход из процедуры, если метод не подходит для решения исходной задачи (а именно, выход на lm происходит в следующих случаях: I) если знаменатель в формулах (2.I3) окажется по абсолютной величине меньше

eps - заданной точности вычислений; 2) если при решении вспомогательных задач Коши окажется $(b-a)/h > 10^{10}$, где h - значение шага интегрирования, полученное к моменту выхода на метку lm), lp - метка, на которую происходит выход из процедуры в случае плохой обусловленности исходной задачи (точнее, если знаменатель в формуле (3.3) окажется по абсолютной величине меньше eps).

§ 4. Вариант ортогональной прогонки для решения общей краевой задачи с вещественными коэффициентами

Метод ортогональной дифференциальной прогонки [II] для решения задачи (I.7) - (I.9) с вещественными коэффициентами в пакете LTPB реализован в двустороннем варианте процедурой aod [27].

4.I. Описание метода

По всему отрезку $[a, b]$ переносим соотношение

$$y \sin\theta(x) + y' \cos\theta(x) = u(x),$$

решая от a к b соответствующие задачи Коши для уравнений ортогональной прогонки:

$$\theta' - \sin^2\theta + p(x)\sin\theta\cos\theta - q(x)\cos^2\theta = 0, \quad x \in [a,b], \tag{4.1}$$

$$\sin\theta(a) = \alpha_1/N_1, \qquad \cos\theta(a) = \beta_1/N_1; \tag{4.2}$$

$$u' + [(q(x)-1)\sin\theta(x)\cos\theta(x) + p(x)\cos^2\theta(x)]u = f(x)\cos\theta(x), \quad x \in [a,b], \tag{4.3}$$

$$u(a) = \gamma_1/N_1, \tag{4.4}$$

где $N_1 = (\alpha_1^2 + \beta_1^2)^{1/2}$. При этом запоминаем величины $\theta_a(x_s)$, $u_a(x_s)$ в точках x_s, $s = 0, \dots, m$, где нас интересует решение исходной задачи (I.7) - (I.9). От b к a решаем уравнения (4.I), (4.3) с граничными условиями

$$\sin\theta(b) = \alpha_2/N_2, \qquad \cos\theta(b) = \beta_2/N_2, \tag{4.5}$$

$$u(b) = \gamma_2/N_2, \tag{4.6}$$

где $N_2 = (\alpha_2^2 + \beta_2^2)^{1/2}$, определяя величины $\theta_b(x_s)$, $u_b(x_s)$, $s = m, m-1, \dots, 0$. Решение задачи (I.7) - (I.9) в точках x_s определяем по формулам

$$y(x_s) = \frac{u_a(x_s)\cos\theta_b(x_s) - u_b(x_s)\cos\theta_a(x_s)}{\Delta(x_s)},$$

$$y'(x_s) = \frac{u_b(x_s)\sin\theta_a(x_s) - u_a(x_s)\sin\theta_b(x_s)}{\Delta(x_s)},$$

где $\Delta(x_s) = \sin\theta_a(x_s)\cos\theta_b(x_s) - \sin\theta_b(x_s)\cos\theta_a(x_s)$,
$s = m, m-1, \dots, 0$.

4.2. Описание процедуры

В процедуре aod [27] численная реализация решения задач Коши (4.I) - (4.4) и (4.I), (4.5), (4.3), (4.6) та же, что и решения соответствующих задач Коши для уравнений классической прогонки в процедуре $clod$ (см. п.2.2).

Имя процедуры и совокупность формальных параметров имеют вид:

$aod(a, b, m, m1, ua, ub, y, y1, pqf, eps, lp)$.

Здесь смысл формальных параметров аналогичен смыслу так же поименованных параметров в процедуре $clod$ (см. п.2.3). Отсутствие идентификатора метки lm в списке формальных параметров процедуры aod обусловлено тем обстоятельством, что теоретически метод [II] обеспечивает "безавостное" решение любой хорошо обусловленной задачи вида (I.7) - (I.9).

§ 5. Вариант ортогональной прогонки для решения общей краевой задачи с комплексными коэффициентами (прогонка Биргера)

Метод ортогональной дифференциальной прогонки [I2] для решения задачи (I.7) - (I.9) с комплексными коэффициентами в пакете LTPBVP реализован в двустороннем варианте процедурой $bicod$ [27].

5.I. Описание метода

По всему отрезку $[a, b]$ переносим соотношение

$$\alpha(x)y + \beta(x)y' = \gamma(x), \tag{5.1}$$

решая от a к b соответствующие задачи Коши для уравнений ортогональной прогонки

$$\alpha' = K(x,\alpha,\beta)\alpha + \beta q(x), \quad x \in [a,b], \tag{5.2}$$

$$\alpha(a) = \alpha_1/N_1; \tag{5.3}$$

$$\beta' = K(x,\alpha,\beta)\beta + \beta p(x) - \alpha, \quad x \in [a,b], \tag{5.4}$$

$$\beta(a) = \beta_1/N_1; \tag{5.5}$$

$$\gamma' = K(x,\alpha,\beta)\gamma + \beta f(x), \quad x \in [a,b], \tag{5.6}$$

$$\gamma(a) = \gamma_1/N_1, \tag{5.7}$$

где $N_1 = (|\alpha_1|^2 + |\beta_1|^2)^{1/2}$.

$$K(x,\alpha,\beta) = \frac{Re\{\alpha\beta^* - q(x)\beta\alpha^* + p(x)\beta\beta^*\}}{\alpha\alpha^* + \beta\beta^*}$$

– вещественная функция (* означает комплексное сопряжение).

Автоматически обеспечивая выполнение условия

$$\alpha(x)\alpha^*(x) + \beta(x)\beta^*(x) = const,$$

для всех x, $x \in [a, b]$, уравнения (5.2), (5.4), (5.6) позволяют устойчиво переносить соотношение (5.I) по всему отрезку $[a, b]$.

При решении задач Коши (5.2) – (5.7) от a к b определяем и храним величины $\alpha_a(x_s)$, $\beta_a(x_s)$, $\gamma_a(x_s)$ в интересующих на точках x_s, $x_s \in [a, b]$, $s = 0, 1, \dots, m$. От b к a решаем уравнения (5.2), (5.4), (5.6) при граничных условиях

$$\alpha(b) = \alpha_2 / N_2, \qquad (5.8)$$

$$\beta(b) = \beta_2 / N_2, \qquad (5.9)$$

$$\gamma(b) = \gamma_2 / N_2, \qquad (5.10)$$

где $N_2 = (|\alpha_2|^2 + |\beta_2|^2)^{1/2}$, определяя величины $\alpha_b(x_s)$, $\beta_b(x_s)$, $\gamma_b(x_s)$, $s = m, m-1, \dots, 0$. Решение задачи (I.7) – (I.9) в точках x_s определяем по формулам

$$y(x_s) = \frac{\gamma_a(x_s)\beta_b(x_s) - \gamma_b(x_s)\beta_a(x_s)}{\Delta(x_s)}, \qquad (5.11)$$

$$y'(x_s) = \frac{\alpha_a(x_s)\gamma_b(x_s) - \alpha_b(x_s)\gamma_a(x_s)}{\Delta(x_s)},$$

где $\Delta(x_s) = \alpha_a(x_s)\beta_b(x_s) - \alpha_b(x_s)\beta_a(x_s)$, $s = m, m-1, \dots, 0$.

5.2. Описание процедуры

В процедуре *bicod* [27] численная реализация задач Коши (5.2) (5.7) и (5.2), (5.8), (5.4), (5.9), (5.6), (5.I0) аналогична численной реализации решения задач Коши для уравнений классической прогонки в процедуре *clod* (см. п.2.2).

Имя процедуры и совокупность формальных параметров имеют вид:

bicod (a, b, m, m1, ua, ub, yr, yi, yr1, yi1, pqf, eps, lp)

Здесь смысл формальных параметров $a, b, m, m1$, *eps* аналогичен смыслу так же поименованных параметров в процедуре *clod* (см. п.2.3). Остальные параметры имеют следующий смысл:

ua – массив *ua*[1:6], задающий граничные условия в точке

(*ua*[1] = Re α_1, *ua*[2] = Im α_1, *ua*[3] = Re β_1, *ua*[4] = Im β_1, *ua*[5] = Re γ_1, *ua*[6] = Im γ_1);

ub – массив *ub*[1:6], задающий граничные условия в точке b (его заполнение аналогично заполнению массива *ua*[1:6]);

yr, *yi*, *yr1*, *yi1* – массивы *yr*[0:m], *yi*[0:m], *yr1*[0:m], *yi1*[0:m] – массивы ответов, соответствующие вычислению в точках x_s комплексного решения задачи (I.7) – (I.9) и его производной (*yr*[s] – значение Re $y(x_s)$; *yi*[s] – значение Im $y(x_s)$; *yr1*[s] – значение Re $y'(x_s)$; *yi1*[s] – значение Im $y'[x_s]$, $s = 0, 1, \dots, m$; *pqf* – идентификатор процедуры с семью параметрами, которая должна быть описана в программе, использующей процедуру *bicod* (процедура *pqf*(x, pr, pi, qr, qi, fr, fi) вычисляет для данного x значения *pr* = Re $p(x)$, *qr* = Re $q(x)$, *pi* = Im $p(x)$, *qi* = Im $q(x)$, *fr* = Re $f(x)$, *fi* = Im $f(x)$ коэффициентов и правой части уравнения (I.7)); *lp* – метка, на которую происходит выход из процедуры в случае плохой обусловленности исходной задачи (точнее, если знаменатель в формулах (5.II) окажется по модулю меньше *eps*).

Отметим, что в списке формальных параметров процедуры *bicod* отсутствует идентификатор метки *lm*, так как теоретически метод [I2] обеспечивает "безавостное" решение любой хорошо обусловленной задачи вида (I.7) – (I.9).

Глава П. Системы дифференциальных уравнений второго порядка

Для решения двухточечных краевых задач для систем о.д.у. второго порядка в пакете *LTPBVP* реализованы следующие методы дифференциальной прогонки: матричная классическая прогонка для решения задачи с действительными коэффициентами и два варианта ортогональной прогонки для решения самосопряженной задачи с действительными коэффициентами и комплексными. Описание и исследование использованных методов см. в [5,I4,I5]. Сообщения об их реализации в *LTPBVP* см. в [28-30].

§ I. Формулировка задачи

Общий вид рассматриваемой задачи:

$$y'' - P(x)\,y = f(x), \qquad x \in [a,b], \tag{I.1}$$

$$A_1 y(a) + B_1 y'(a) = C_1, \tag{I.2}$$

$$A_2 y(b) + B_2 y'(b) = C_2. \tag{I.3}$$

Здесь n есть заданное целое положительное число - порядок системы; $P(x)$ - заданная при $x \in [a,b]$ квадратная матрица порядка n; $f(x)$ - заданный при $x \in [a,b]$ столбец высоты n; A_i, B_i $(i=1,2)$ - заданные матрицы порядка n такие, что прямоугольные матрицы $\|A_1 \vdots B_1\|$, $\|A_2 \vdots B_2\|$ имеют ранг n; C_1, C_2 - заданные столбцы высоты n.

Далее предполагаем, что функции $P(x)$ и $f(x)$ непрерывны на $[a,b]$.

Для решения задачи (I.I) - (I.3) с действительными коэффициентами при условии

$$B_1 = B_2 = E \tag{I.4}$$

использовалась матричная классическая прогонка (здесь и далее E - единичная матрица). Описание метода приведено в [5].

Достаточными условиями устойчивости метода, по-видимому, являются следующие: $P(x)$ - симметрическая положительно-определенная матрица; A_1, A_2 - симметрические положительно-полуопределенные матрицы. В пакете $LTPBVP$ матричная классическая прогонка реализована на языке АЛГОЛ в виде процедуры $cl32d$. Сообщение о реализации алгоритма см. в [28].

Для решения самосопряженной задачи (I.I) - (I.3) для с.о.д.у. с действительными коэффициентами, т.е. при условиях

$$\begin{aligned} P &= P^*, \\ A_1 B_1^* &= B_1 A_1^*, \\ A_2 B_2^* &= B_2 A_2^*, \end{aligned} \tag{I.5}$$

применяется вариант ортогональной прогонки Барретта (описание метода см. в [I4]), основанный на преобразовании Прюфера для матричных о.д.у. При этом предполагается, что в дополнение к условиям (I.5) матрицы A_1, A_2, B_1, B_2 удовлетворяют условиям нормировки:

$$\begin{aligned} A_1 A_1^* + B_1 B_1^* &= E, \\ A_2 A_2^* + B_2 B_2^* &= E. \end{aligned} \tag{I.6}$$

Используя подход работы [II], можно показать, что метод [I4] устойчив настолько, насколько хорошо обусловлена исходная краевая задача. В пакете $LTPBVP$ этот вариант прогонки реализован на языке АЛГОЛ процедурой $bts2d$. Сообщение о реализации алгоритма см. в [29].

Для случая самосопряженной задачи (I.I) - (I.3) для систем о.д.у. с комплексными коэффициентами, удовлетворяющими условию (I.5), использовался метод ортогональной прогонки Лидского-Нейгауз, приведенный в [I5]. В [I5] показано, что этот метод устойчив, если исходная краевая задача устойчива относительно малых изменений коэффициентов, входящих в уравнение, и относительно малых изменений коэффициентов в граничных условиях. В пакете $LTPBVP$ этот вариант прогонки реализован на языке АЛГОЛ процедурой $lns2d$ (см. [30]).

§ 2. Матричная классическая прогонка

2.I. Описание метода

Алгоритм метода, приведенного в [5], для решения задачи (I.I) - (I.3) при условиях (I.4), реализуемый процедурой $cls2d$, состоит в следующем. По всему отрезку $[a,b]$ переносим соотношение

$$y' = \alpha(x)\,y + \beta(x),$$

решая соответствующие задачи Коши для уравнений классической прогонки:

$$\begin{aligned} \alpha' + \alpha^2 - P(x) &= 0, \\ \beta' + \alpha(x)\,\beta &= f(x), \qquad x \in [a,b]. \end{aligned} \tag{2.I}$$

Здесь $\alpha(x)$ есть матрица порядка n; $\beta(x)$ - столбец высоты n.

От a к b решаем систему (2.I) при начальных условиях:

$$\alpha(a) = -A_1, \qquad \beta(a) = C_1. \tag{2.2}$$

При этом запоминаем значения $\alpha_a(x)$ и $\beta_a(x)$ в тех точках x_s, $s = 0,1,\ldots,m$, отрезка $[a,b]$, где нас интересует решение исходной задачи.

От b к a интегрируем уравнения (2.I) с условиями

$$\alpha(b) = -A_2, \qquad \beta(b) = C_2, \tag{2.3}$$

вычисляя $\alpha_b(x_s)$, $\beta_b(x_s)$, $s = m, m-1, \ldots, 0$.

Решение $y(x)$ исходной задачи и его производную определяем в точках x_s из системы линейных алгебраических уравнений

$$(\alpha_a - \alpha_b)\, y(x_s) = \beta_b - \beta_a \ , \qquad (2.4)$$
$$y'(x_s) = \alpha_a\, y(x_s) + \beta_a \ ,$$
$$s = m,\ m-1,\ \ldots,\ 0 \ .$$

2.2. Численная реализация метода

Отметим, что здесь используется "двусторонний" вариант прогонки в применении к решению систем о.д.у. второго порядка. Этот прием позволяет хранить промежуточную информацию только в тех точках заданного отрезка, в которых отыскивается решение исходной задачи.

Предлагаемый алгоритм матричной классической прогонки дает решение исходной задачи в точках $x_s = a + s \times (b-a)/m$, $s = 0, 1, \ldots, m$. Граничное условие в точке $a\,(a = x_0)$ переносится в точку x_1, затем из x_1 в x_2 и т.д. $(x_m = b)$, так же переносится в каждую из точек x_s, $s = m-1, \ldots, 0$, граничное условие в точке b. Для дальнейших вычислений запоминаются только результаты переноса условия (2.2) в точки x_s. Никакие результаты, получаемые во внутренних точках интервалов (x_s, x_{s+1}), не запоминаются.

Численное решение задачи Коши на каждом из отрезков $[x_s, x_{s+1}]$ (соответственно, $[x_{s+1}, x_s]$) осуществляется методом Рунге-Кутта четвертого порядка точности с автоматическим выбором шага. Для интегрирования на один шаг используется процедура RK1ST, приведенная в [47]. Контроль точности и выбор шага в предлагаемой процедуре производятся так же, как в п.2.2 главы I с естественным обобщением на системы уравнений.

Линейная алгебраическая система (2.4) решается методом Гаусса с выбором главного элемента по строке.

2.3. Описание процедуры

Имя процедуры и совокупность формальных параметров имеют вид:

cls2d (a, b, m, m1, n, ua, ub, y, y1, pf, eps, lm, lp).

Формальные параметры означают следующее:

a, b – концы интервала интегрирования;

m – целое положительное число точек x_s, в которых будут получены ответы; значения $y(x)$ и $y'(x)$ вычисляются в $m+1$ точках x_s, $x_s = a + s \times (b-a)/m$, $s = 0, 1, \ldots, m$;

m1 – целочисленный параметр, $m1 = 2^k$ ($0 \le k$ – целое число), который задает начальный шаг интегрирования h, $h = (b-a)/(m \times m1)$, при решении задачи Коши;

n – целое положительное число – порядок системы (I.I);

ua – массив, $ua[1:n, 1:n+1]$ – расширенная матрица граничных условий в точке a, $ua = \|-A_1 \vdots C_1\|$;

ub – массив, $ub[1:n, 1:n+1]$ – расширенная матрица граничных условий в точке b, $ub = \|-A_2 \vdots C_2\|$;

y – массив, $y[0:m, 1:n]$ – массив ответов, $y[s, i]$ – значение $y_i(x_s)$;

y1 – массив, $y1[0:m, 1:n]$ – массив значений производной искомой функции; $y1[s, i]$ – значение $y_i'(x_s)$;

pf – имя процедуры с тремя параметрами, которая должна быть описана в программе, использующей рассматриваемую процедуру. Процедура $pf(x, p, f)$ вычисляет для данного x значения, заполняющие массивы $p[1:n, 1:n]$ (коэффициенты уравнения (I.I)) и $f[1:n]$ (правые части уравнения (I.I));

eps – заданная абсолютная точность вычислений; интегрирование задачи Коши на один шаг осуществляется с точностью $eps1 = \frac{eps}{m \times m1}$;

lm – метка, на которую передается управление в случае, когда данный метод не подходит для решения исходной задачи, именно когда $(b-a)/h > 10^{10}$ (здесь h – значение шага интегрирования, полученное к моменту выхода на метку lm);

lp – метка, на которую передается управление при условии, если исходная задача плохо обусловлена, а именно когда при решении системы (2.5) методом Гаусса очередной главный элемент станет по модулю меньше eps.

§ 3. Вариант ортогональной прогонки Барретта для самосопряженной задачи с действительными коэффициентами

3.I. Описание метода

Алгоритм метода, предложенного в [I4] для решения задачи (I.I) – (I.3) при выполнении условий (I.5), оформлен в пакете в виде процедуры bts2d [29] и состоит в следующем. Пусть выполнены условия (I.5) и пусть матрицы A_1, A_2, B_1, B_2 удовлетворяют условиям нормировки (I.6). Отметим, что на основании леммы I.I в [I4] из (I.5), (I.6) будет следовать также выполнение условий:

$$A_1^* B_1 = B_1^* A_1 \ , \quad A_2^* B_2 = B_2^* A_2 \ ,$$
$$A_1^* A_1 + B_1^* B_1 = E \ , \quad B_2^* B_2 + A_2^* A_2 = E \ .$$

По всему отрезку $[a, b]$ переносим соотношение

$$S(x)\, y' + C(x)\, y = \gamma(x) \ ,$$

требуя для всех x выполнения условий

$$SS^* + CC^* = E\,, \quad SC^* = CS^*\,, \tag{3.2}$$

$$S^*S + C^*C = E\,, \quad S^*C = C^*S\,. \tag{3.3}$$

Матричные функции $S(x)$ и $C(x)$ и вектор-функцию $\gamma(x)$ определяем из решения задач Коши для уравнений

$$S' = -Q\cdot(x,S,C)\,C\,,$$

$$C' = Q(x,S,C)\,S\,, \tag{3.4}$$

$$\gamma' = (SC^* + SPC^*)\gamma + Sf\,, \tag{3.5}$$

где $Q(x,S,C) = CC^* - SPS^*$.

Нетрудно проверить, что для уравнений (3.4) выполняются условия (3.3) для всех x, $x \in [a,b]$. Из выполнения соотношений (3.3) следует также выполнение условий (3.2) (см. [I4]). Выполнение условий (3.2) и (3.3) обеспечивает устойчивое интегрирование уравнений (3.4).

От a к b решаем систему (3.4) - (3.5) при начальных условиях:

$$S(a) = B_1\,, \quad C(a) = A_1\,, \quad \gamma(a) = C_1\,.$$

При этом запоминаем значения $S_a(x_j)$, $C_a(x_j)$ и $\gamma_a(x_j)$ в точках x_j, $j = 0,1,\dots,m$, отрезка $[a,b]$, в которых будут получены ответы. От b к a система (3.4) - (3.5) интегрируется при условиях

$$S(b) = B_2\,, \quad C(b) = A_2\,, \quad \gamma(b) = C_2\,.$$

Здесь вычисляются значения $S_b(x_j)$, $C_b(x_j)$ и $\gamma_b(x_j)$.

Решение $y(x)$ исходной задачи в точках x_j, $j = 0,1,\dots,m$, и его производную $y'(x)$ определяем из системы линейных алгебраических уравнений:

$$S_a(x_j)\,y'(x_j) + C_a(x_j)\,y(x_j) = \gamma_a(x_j)\,,$$

$$S_b(x_j)\,y'(x_j) + C_b(x_j)\,y(x_j) = \gamma_b(x_j)\,, \tag{3.6}$$

$$j = m,\ m-1,\ \dots,\ 0\,.$$

Отметим, что программа не производит автоматического нормирования граничных матриц с тем, чтобы выполнялись условия (I.6), так что перед обращением к процедуре $bts2d$ следует предварительно привести эти матрицы к необходимому виду. Это можно сделать, например, следующим образом. Пусть

$$L = A_1 A_1^* + B_1 B_1^*\,. \tag{3.7}$$

Заметим, что $L^* = L$ и $L > 0$. Заменим условие (I.2) эквивалентным ему условием:

$$TA_1\,y(a) + TB_1\,y'(a) = TC_1\,.$$

Тогда (3.7) запишется в виде:

$$T(A_1A_1^* + B_1B_1^*)\,T^* = TLT^*\,.$$

Положим $T = MV$, где V - унитарная матрица, приводящая L к диагональному виду:

$$VLV^* = \Lambda = \begin{Vmatrix} \lambda_1 & \dots & 0 \\ 0 & \dots & \lambda_n \end{Vmatrix}, \quad \lambda_j > 0,\ j = 1,\dots,n\,; \tag{3.8}$$

M - диагональная матрица:

$$M = (\sqrt{\Lambda})^{-1} = \begin{Vmatrix} (\sqrt{\lambda_1})^{-1} & \dots & 0 \\ 0 & \dots & (\sqrt{\lambda_n})^{-1} \end{Vmatrix}\,.$$

Преобразование (3.8) осуществляется, например, методом вращений Якоби. Нетрудно видеть, что новые матрицы $\tilde{A}_1 = TA_1$ и $\tilde{A}_2 = TB_1$ уже будут удовлетворять условиям (I.6).

3.2. Описание процедуры

Все сказанное в п. 2.2 относительно численной реализации матричной классической прогонки остается верным без каких-либо дополнений и для рассматриваемого здесь варианта ортогональной прогонки.

Имя процедуры и совокупность формальных параметров имеют вид:

$bts2d(a, b, m, m1, n, ua, ub, y, y1, pf, eps, em, ep)$.

Все формальные параметры означают здесь то же, что и для матричной классической прогонки. Массивы граничных условий ua, ub в этом случае задаются следующим образом: $ua[1{:}n,\ 1{:}2\times n+1]$, $ub[1{:}n,\ 1{:}2\times n+1]$ - расширенные матрицы граничных условий в точках a и b соответственно;

$$ua = \|A_1 \vdots B_1 \vdots C_1\|\,, \qquad ub = \|A_2 \vdots B_2 \vdots C_2\|\,.$$

§ 4. Вариант ортогональной прогонки Лидского-Нейгауз для самосопряженной задачи с комплексными коэффициентами

4.I. Описание метода

Изложенный здесь алгоритм метода, предложенного в [I5] для решения самосопряженной задачи (I.1) - (I.3) с комплексными коэффициента-

ми при выполнении условий (I.5), оформлен в LTPBVP в виде процедуры lns2d [30].

Пусть для задачи (I.I) – (I.3) выполнены условия (I.5). Алгоритм состоит в следующем. По всему отрезку $[a,b]$ переносим соотношение

$$\tfrac{1}{2}(E-u(x))y'(x)-\tfrac{i}{2}(E+u(x))y(x)=\xi(x)\ , \qquad (4.1)$$

решая для $u(x)$ и $\xi(x)$ соответствующие задачи Коши для уравнений ортогональной прогонки:

$$u'=-i\{E+\tfrac{1}{2}(u+u^*)-\tfrac{1}{2}(E-u)P(E-u^*)\}u\ ,$$
$$\xi'=\tfrac{i}{2}\{(E-u)P-(E+u)\}\xi+\tfrac{1}{2}(E+u)f\ . \qquad (4.2)$$

От a к b решаем уравнения (4.2) при условиях:

$$u(a)=(A_1-iB_1)^{-1}(A_1+iB_1)\ ,$$
$$\xi(a)=(B_1+iA_1)^{-1}C_1\ .$$

От b к a решаем систему (4.2) при условиях:

$$u(b)=(A_2-iB_2)^{-1}(A_2+iB_2)\ ,$$
$$\xi(b)=(B_2+iA_2)^{-1}C_2\ .$$

Решение $y(x)$ исходной задачи и его производную $y'(x)$ в каждой точке x_s, $s=0,1,\dots,m$, отрезка $[a,b]$ определяем из системы линейных алгебраических уравнений:

$$\tfrac{1}{2}(E-u_a(x_s))y'(x_s)-\tfrac{i}{2}(E+u_a(x_s))y(x_s)=\xi_a(x_s)\ ,$$
$$\tfrac{1}{2}(E-u_b(x_s))y'(x_s)-\tfrac{i}{2}(E+u_b(x_s))y(x_s)=\xi_b(x_s)\ , \qquad (4.3)$$
$$s=m,\ m-1,\dots,0\ .$$

Отметим, что для функции $u(x)$, удовлетворяющей первому из уравнений (4.2), для каждого $x\in[a,b]$ выполняется условие $u^*(x)\ u(x)=E$, что обеспечивает устойчивое решение соответствующих задач Коши.

4.2. Численная реализация метода

Для изложенного здесь метода алгоритм, определенный в LTPBVP в виде процедуры lns2d, дает решение исходной задачи для произвольного набора точек $x_0,\dots,x_m$, $a=x_0<x_1<\dots<x_m=b$

или $a=x_0>x_1>\dots>x_m=b$. Граничное условие в точке a, $a=x_0$, переносится в точку x_1, затем из x_1 в x_2 и т.д. до точки b, $x_m=b$, так же переносится в каждую из точек x_s, $s=m-1,\dots,0$, граничное условие в точке b, $x_m=b$. При этом величины, получаемые в результате прямой прогонки, запоминаются только в тех точках, в которых требуется получить ответ.

Численное решение задачи Коши на каждом из отрезков $[x_s,x_{s+1}]$, $s=0,1,\dots,m-1$ (соответственно на $[x_{s+1},x_s]$, $s=m-1,m-2,\dots,0$) осуществляется методом Рунге-Кутта четвертого порядка точности с автоматическим выбором шага. Для интегрирования на один шаг используется процедура RK2VH, сходная с процедурой RK1ST, приведенной в [47]. Контроль точности и выбор шага осуществляются следующим образом (далее в качестве иллюстрации рассматривается только перенос условия, заданного в точке a). Разобьем отрезок интегрирования $[x_s,x_{s+1}]$ на I0 равных частей и положим $\hat{x}_i=x_s+i\times h$, где $h=(x_{s+1}-x_s)/10$. Вычислим по формуле Рунге-Кутта $y_h(\hat{x}_i)$, $y_{h/2}(\hat{x}_i)$, $y'_h(\hat{x}_i)$ $y'_{h/2}(\hat{x}_i)$. Здесь $y'_h(\hat{x}_i)$, $y'_h(\hat{x}_i)$ $y_{h/2}(\hat{x}_i)$, $y'_{h/2}(\hat{x}_i)$ – массивы значений искомой функции и ее производной в точке $\hat{x}_i$ при счете с шагом h и соответственно при счете с шагом $h/2$. Шаг интегрирования выбирается таким образом, чтобы в точке $\hat{x}_i$ выполнялись условия:

$$|y_h(\hat{x}_i)-y_{h/2}(\hat{x}_i)|<eps\times|h|\ ,$$
$$|y'_h(\hat{x}_i)-y'_{h/2}(\hat{x}_i)|<eps\times|h|\ . \qquad (4.4)$$

Здесь eps – заданная точность вычислений. Если условия (4.4) не выполняются, то шаг интегрирования делится пополам, в противном случае шаг интегрирования удваивается. Интегрирование от точки $\hat{x}_i$ до точки $\hat{x}_{i+1}=\hat{x}_i+h$ выполняется с тем наибольшим шагом h, при котором выполнены условия (4.4). Затем проверяется выполнение условий (4.4) в точке $\hat{x}_{i+1}$ и т.д.

Линейная алгебраическая система (4.3) решается методом Гаусса с выбором главного элемента по строке.

Для обращения матриц $\|A_1-iB_1\|$, $\|B_1+iA_1\|$, $\|A_2-iB_2\|$, $\|B_2+iA_2\|$ применяется процедура INVERT2, в которой использован метод Гаусса.

4.3. Описание процедуры

Имя процедуры и совокупность формальных параметров имеют вид:

$lns2d(x, m, n, ua, ub, yr, yi, yr1, yi1, pf, eps, lm, lp)$.

Формальные параметры обозначают следующее:

x – **массив,** $x[0:m]$ – **массив, содержащий значения независимой переменной, для которых требуется найти решение;**

смысл параметров m , n **тот же, что в § I;**

ua, ub – **массивы,** $ua[1:n, 1:4\times n+2]$, $ub[1:n, 1:4\times n+2]$ – **расширенные матрицы граничных условий в точках** a **и** b **соответственно;**

$ua = \| Re\,B_1 \vdots Im\,B_1 \vdots Re\,A_1 \vdots Im\,A_1 \vdots Re\,C_1 \vdots Im\,C_1 \|$,

$ub = \| Re\,B_2 \vdots Im\,B_2 \vdots Re\,A_2 \vdots Im\,A_2 \vdots Re\,C_2 \vdots Im\,C_2 \|$;

yr, yi – **массивы,** $yr[1:n, 0:m]$, $yi[1:n, 0:m]$ – **массивы ответов;** $yr[j, s]$ **и** $yi[j, s]$ **– значения** $Re\,y_j(x_s)$ **и** $Im\,y_j(x_s)$ **соответственно;**

$yr1$, $yi1$ – **массивы,** $yr1[1:n, 0:m]$, $yi1[1:n, 0:m]$ – **массивы ответов,** $yr1[j, s]$ **и** $yi1[j, s]$ **– значения** $Re\,y'_j(x_s)$ **и** $Im\,y'_j(x_s)$ **соответственно;**

pf – **идентификатор процедуры с пятью параметрами, которая должна быть описана в программе; процедура** $pf(tau, p1, p2, f1, f2)$ **для заданного независимого переменного** tau **вычисляет значения, заполняющие массивы** $p1[1:n, 0:m]$, $p2[1:n, 0:m]$ (коэффициенты исходного уравнения (I.I), $P(x) = p1 + i \times p2$), **и массивы** $f1[1:n]$, $f2[1:n]$ (правые части уравнения (I.I), $f(x) = f1 + i \times f2$);

eps – **положительное число, регулирующее течение процесса вычислений;**

lm – **метка, на которую передается управление в случае, когда при обращении матриц** $\| A_1 - iB_1 \|$, $\| A_2 - iB_2 \|$, $\| B_1 + iA_1 \|$ **или матрицы** $\| B_2 + iA_2 \|$ **методом Гаусса окажется, что модуль очередного главного элемента меньше** eps ;

lp – **метка, на которую передается управление при условии, если исходная задача плохо обусловлена, точнее, когда при решении системы (4.3) методом Гаусса очередной главный элемент станет по модулю меньше** eps ; **кроме того, на** lp **передается управление и тогда, когда** $h < 10^{-10}$, **где** h – **значение шага интегрирования, полученные к моменту выхода на метку** lp .

Глава III. Системы дифференциальных уравнений первого порядка

Для систем дифференциальных уравнений первого порядка в пакете LTPBVP рассмотрены два типа краевых задач: двухточечная краевая задача с разделенными условиями и общая двухточечная краевая задача. Для решения первой из этих задач реализованы следующие методы: классическая прогонка [I6,I7] и четыре варианта ортогональной прогонки [8,I8-20] (методы, предложенные в работах [I8] и [I9], метод Бахвалова [8, стр.58I-583], метод Годунова [20]). Для решения второй из этих задач реализован прием Мошинского [2I]. Всюду рассматриваются только вещественные системы о.д.у. и вещественные краевые условия.

Описание и исследование использованных методов см. также в [48, 49]; сообщения об их реализации в пакете LTPBVP см.в [3I-36].

§ I. Формулировка задач

I.I. Рассмотрим систему о.д.у.

$$y' - P(x)\,y = f(x) , \quad x \in [a, b] . \tag{I.I}$$

Здесь a и b – заданные числа; $P(x)$ – заданная при $x \in [a,b]$ квадратная матрица порядка n ; $f(x)$ – заданный при $x \in [a, b]$ столбец высоты n ; n – заданное целое положительное число. Далее предполагаем, что функции $P(x)$ и $f(x)$ непрерывны на $[a, b]$. Эта система о.д.у. дополняется одним из следующих краевых условий.

I.2. Разделенные краевые условия

$$\psi_a\,y(a) = g_a , \tag{I.2}$$

$$\psi_b\,y(b) = g_b . \tag{I.3}$$

Здесь ψ_a – заданная матрица ранга k_a , имеющая k_a строк и n столбцов; ψ_b – заданная матрица ранга $n-k_a$, имеющая $n-k_a$ строк и n столбцов; g_a и g_b – заданные столбцы высоты соответственно k_a и $n-k_a$; k_a – заданное целое положительное число, меньшее n .

I.3. Общая двухточечная краевая задача:

$$Ay(a) + By(b) = C , \tag{I.4}$$

где A и B – заданные квадратные матрицы порядка n , ранг матрицы $\| A \vdots B \|$ равен n , C – заданный столбец высоты n .

§ 2. Ортогональные прогонки

Методы ортогональной прогонки обеспечивают (теоретически) "безавостность" алгоритма в тех случаях, когда исходная краевая задача является невырожденной.

2.I. Описание методов

2.I.I. Методы, предложенные в [I8] и [I9], реализованы на АЛГОЛе соответственно процедурами $as1d$ **и** $us1d$ **. Эти методы состоят в следующем.**

Каждое из условий (I.2) и (I.3) переносится в точки $x_0, x_1, \dots, x_m$. Этот перенос осуществляется решением на $[a, b]$ задач Коши:

$$\varphi_a' + \varphi_a P - S\varphi_a = 0 , \qquad \varphi_a(a) = \psi_a ,$$
$$\gamma_a' - S\gamma_a = \varphi_a f , \qquad \gamma_a(a) = g_a ,$$
$$\varphi_b' + \varphi_b P - S\varphi_b = 0 , \qquad \varphi_b(b) = \psi_b ,$$
$$\gamma_b' - S\gamma_b = \varphi_b f , \qquad \gamma_b(b) = g_b .$$

Здесь $S(x)$ - матричная функция, характеризующая метод. Для метода, предложенного в [I8], $S = \varphi P \varphi^* (\varphi\varphi^*)^{-1}$; следствием этого являются соотношения

$$\varphi_a \varphi_a^* = const , \quad \varphi_b \varphi_b^* = const . \qquad (2.1)$$

Для метода, предложенного в [I9], $S = (\varphi P \varphi^* - \varphi f \gamma^*)(\varphi\varphi^* + \gamma\gamma^*)^{-1}$,

следствием этого являются соотношения

$$\varphi_a \varphi_a^* + \gamma_a \gamma_a^* = const , \quad \varphi_b \varphi_b^* + \gamma_b \gamma_b^* = const . \qquad (2.2)$$

Для каждой из точек x_s $(s = 0, \dots, m)$ составляется система уравнений

$$\varphi_a(x_s) y(x_s) = \gamma_a(x_s) ,$$
$$\varphi_b(x_s) y(x_s) = \gamma_b(x_s) , \qquad (2.3)$$

из которых и находится $y(x_s)$.

2.I.2. Методы Бахвалова (см. [8]) и Годунова (см. [20]) реализованы на АЛГОЛе соответственно процедурами $bas1d$ **и** $gs1d$ **. Изложим сначала свойства, которые являются общими для обоих методов.**

Возьмем на $[a, b]$ любую точку c . Перенесем граничное условие (I.2) по всему отрезку $[a, c]$, записывая соотношение, получающиеся в точке x, $x \in [a, c]$, в виде

$$y = Z z + u , \qquad (2.4)$$

где Z - матрица ранга $n - k_a$, имеющая $n - k_a$ столбцов и n строк, u - столбец высоты n, z - столбец высоты $n - k_a$. Здесь подпространство, порожденное столбцами Z, и вектор $u(x)$, с точностью до слагаемого в указанном подпространстве, определяются уравнением (I.I) и граничным условием (I.2); $z(x)$ определяется выбором конкретных матриц $Z(x)$ и $u(x)$ и выбором конкретного решения $y(x)$.

Граничное условие (I.2) также приводим к виду

$$y(a) = Z_0 z + u_0 ,$$

где u_0 - какое-либо частное решение системы линейных алгебраических уравнений

$$\psi_a u = \gamma_a ,$$

а столбцы Z_0 - какая-либо фундаментальная система решений соответствующей однородной системы уравнений.

В каждом из рассматриваемых двух методов $Z(x)$ и $u(x)$ получаются решением от a к c некоторых задач Коши (так называемая прямая прогонка). Для $z(x)$ в каждом из методов получается дифференциальное уравнение.

Аналогично в форме

$$y = \bar{Z} \bar{z} + \bar{u} \qquad (2.5)$$

перенесем по отрезку $[b, c]$ условие (I.3).

Используя вычисленные Z , u , $\bar{Z}$, $\bar{u}$, составим в точке c систему линейных алгебраических уравнений

$$Z(c) z_c + u(c) = \bar{Z}(c) \bar{z}_c + \bar{u}(c) , \qquad (2.6)$$

из которой найдем z_c и $\bar{z}_c$.

Используя эти значения и упомянутые о.д.у. для z и $\bar{z}$, решим (от c к a для z и от c к b для $\bar{z}$) возникающие задачи Коши (так называемая обратная прогонка).

Значения y в нужных точках вычислим по формулам (2.4) (для $[a, c]$) и (2.5) (для $[b, c]$).

Конкретизируем теперь формулы, соответствующие каждому из рассматриваемых методов. Эти формулы мы выпишем для отрезка $[a, c]$; для $[b, c]$ формулы совершенно аналогичны с очевидной заменой a ,

Z, u, r, k_a на b, $\bar{Z}$, $\bar{u}$, $\bar{r}$, $n-k_a$.

2.I.2.I. В методе Бахвалова $Z(x)$ и $u(x)$ – решения задач Коши

$$Z' = (E - Z(Z^*Z)^{-1}Z^*)PZ, \quad Z(a) = Z_0,$$

$$u' = (E - Z(Z^*Z)^{-1}Z^*)(Pu + f), \quad u(a) = u_0.$$

Следствием первого из этих уравнений является соотношение $Z^*Z = const$ Уравнение для r

$$r' = (Z^*Z)^{-1}Z^*(PZr + Pu + f)$$

и начальное условие, как уже говорилось, $r(c) = r_c$.

2.I.2.2. В методе Годунова мы разбиваем отрезок $[a, c]$ на m_1 участков точками $\hat{x}_1, \dots, \hat{x}_{m_1-1}$, $a < \hat{x}_1 < \dots < \hat{x}_{m_1-1} < c$ (для простоты изложения предполагаем, что $a < b$), положим дополнительно $\hat{x}_0 = a$, $\hat{x}_{m_1} = c$. Полагаем $\hat{Z}_0 = Z_0$, $\hat{u}_0 = u_0$ и для $P = 0, \dots, m_1 - 1$ делаем следующее.

Отнормируем столбцы $\hat{Z}_p$. Вычитаем из $\hat{u}_p$ его ортогональную проекцию на подпространство, порожденное столбцами $\hat{Z}_p$. Решаем на отрезке $[\hat{x}_p, \hat{x}_{p+1}]$ задачи Коши

$$Z' - PZ = 0, \quad Z(\hat{x}_p) = \hat{Z}_p,$$

$$u' - Pu = f, \quad u(\hat{x}_p) = \hat{u}_p.$$

Для обратной прогонки в этом методе $r = const$ на каждом интервале $(\hat{x}_{p+1}, \hat{x}_p)$. При переходе от отрезка $[\hat{x}_p, \hat{x}_{p+1}]$ к отрезку $[\hat{x}_{p-1}, \hat{x}_p]$ r меняется так, чтобы $Zr + u$ было непрерывным в точке $\hat{x}_p$. Если при переходе через $\hat{x}_p$ в прямой прогонке было взято

$$Z(\hat{x}_p + 0) = Z(\hat{x}_p - 0)\Omega_p,$$

$$u(\hat{x}_p + 0) = u(\hat{x}_p - 0) - Z(\hat{x}_p - 0)d_p$$

(Ω_p – квадратная матрица порядка $n - k_a$, d_p – столбец высоты $n - k_a$), то

$$r(\hat{x}_p - 0) = \Omega_p r(\hat{x}_p + 0) - d_p.$$

2.2. Численная реализация методов

2.2.I. Для методов, предложенных в [I8] и [I9], алгоритмы *LTPBVP* дают решение в точках x_s,

$$x_s = a + s(b-a)/m, \qquad s = 0, \dots, m$$

Граничное условие в точке a ($a = x_0$) переносится в точку x_1, затем из x_1 в x_2, ..., из x_{m-1} в x_m ($x_m = b$). Аналогично переносится в каждую из точек x_s ($s = m-1, \dots, 0$) граничное условие в точке b. Для дальнейших вычислений запоминаются только или результаты переноса условия (I.2) в точки x_s (при $k_a \le n/2$) или результаты переноса условия (I.3) в те же точки (при $k_a > n/2$); никакие результаты, получаемые во внутренних точках интервалов (x_s, x_{s+1}), не запоминаются.

Численное решение задачи Коши на каждом из отрезков $[x_s, x_{s+1}]$ (соответственно, $[x_{s+1}, x_s]$) осуществляется с помощью процедуры *transfer* [50] (здесь эта процедура несущественно изменена). В этой процедуре использован метод Рунге-Кутта четвертого порядка точности с автоматическим выбором шага. Контроль точности и выбор шага осуществляются с помощью соотношений (2.I, 2.2) следующим образом (далее для краткости рассматривается только перенос условия, заданного в a). Разобьем отрезок интегрирования $[x_s, x_{s+1}]$ на m_1 частей и положим $\hat{x}_i = x_s + iH$ $(i = 0, \dots, m_1)$, где $H = (x_{s+1} - x_s)/m_1$. Шаг интегрирования выбирается так, чтобы в точках $\hat{x}_i$ выполнялось условие:

$$|\varphi_a(\hat{x}_i)\varphi_a^*(\hat{x}_i) - \varphi_a(x_s)\varphi_a^*(x_s)| \le \frac{i}{m_1}\,eps, \qquad (2.7)$$

– для метода, предложенного в [I8], $i = 1, \dots, m_1$,

$$|\varphi_a(\hat{x}_i)\varphi_a^*(\hat{x}_i) + \gamma_a(\hat{x}_i)\gamma_a^*(\hat{x}_i) - \varphi_a(x_s)\varphi_a^*(x_s) - \gamma_a(x_s)\gamma_a^*(x_s)| \le \frac{i}{m_1}\,eps, \qquad (2.8)$$

– для метода, предложенного в [I9]. $i = 1, \dots, m_1$,

Здесь eps – заданная точность вычислений. На каждом из отрезков $[\hat{x}_{i-1}, \hat{x}_i]$ интегрирование ведется с постоянным шагом h. Если в точке $\hat{x}_i$ условия (2.7, 2.8) не выполняются, то интегрирование по этому отрезку повторяется с шагом вдвое меньшим, чем h, и т.д. На отрезке $[\hat{x}_0, \hat{x}_1]$ первое интегрирование проводится с шагом $h = H/m_2$. На каждом последующем отрезке начальный шаг выбирается в зависимости от того, при каких шагах выполнялись условия (2.7,2.8) на предыдущих отрезках. Числа m_1 и m_2 задаются пользователем; очевидно, что время счета процедуры *transfer* может существенно зависеть от выбора значений m_1 и m_2.

Обращение симметрической положительно-определенной матрицы (это

матрица $\varphi_a\ \varphi_a^*$ для одного метода и матрица $\varphi_a\varphi_a^* + \gamma_a\gamma_a^*$ для другого) осуществляется упрощенным вариантом метода квадратных корней на основе алгоритма 66 [51].

Для решения систем линейных алгебраических уравнений (2.3) используется процедура $gauss$, реализующая метод Гаусса с выбором главных элементов по строке.

2.2.2. Для метода Бахвалова алгоритм, реализованный в $LTPBVP$, дает решение в точках x_s,

$$x_s = a + s(c-a)/m_a, \qquad s = 0, \dots, m_a$$

и

$$x_s = b + s(c-b)/m_b, \qquad s = 0, \dots, m_b - 1 .$$

Далее для простоты изложения рассмотрим только решение на отрезке $[a, c]$.

Матрица Z_0 вычисляется процедурой так, чтобы $Z_0^* Z_0 = E$; тем самым $Z^*(x)\, Z(x) = E$ для всех x .

Численное решение задач Коши в процедуре осуществляется по усовершенствованному методу Эйлера (см. [8, стр.451]); шаг интегрирования выбирается постоянным: $h_a = \frac{c-a}{m_1 \times m_a}$, где целое положительное число m_1 задается пользователем.

Величины, получаемые в результате прямой прогонки, запоминаются только в точках, в которых нужно получить ответ; нужные в промежуточных точках для обратной прогонки результаты прямой прогонки перевычисляются с использованием приема, предложенного в [52].

Для решения систем линейных алгебраических уравнений при определении начальных значений Z_0 , z_c , а также для обращения матрицы $Z^* Z$ применен видоизмененный в каждом из этих случаев соответствующим образом алгоритм Гаусса-Жордана с выбором главных элементов по строке.

2.2.3. Метод Годунова реализован для случая, когда $c = b$ (тем самым надобность расчетов на отрезке $[b, c]$ отпадает) и для произвольного набора $x_0, \dots, x_m$ $(a = x_0 < x_1 < \dots < x_m = b$ или $a = x_0 > x_1 > \dots > x_m = b)$. Точки ортогонализации $\hat{x}_0, \hat{x}_1, \dots, \hat{x}_{m_1}$ выбираются равномерно на $[a, b]$.

Процедура самостоятельно формирует Z_0 и u_0 .

Интегрирование вспомогательных задач Коши между точками ортогонализации проводится с помощью процедуры $urkvh$ [53], использующей метод Рунге-Кутта четвертого порядка точности с автоматическим выбором шага в зависимости от заданной точности eps . Начальный

шаг берется равным $(b-a)/(m_1 \times m_2)$, где целое положительное число m_2 задается пользователем.

Ортогонализация векторов производится по методу Шмидта (см., например, [8, стр.80]). В этом случае матрица Ω_ρ треугольная. Возникающие в процедуре системы линейных алгебраических уравнений решаются методом Гаусса с выбором главных элементов по строке [54].

Матрицы Ω_ρ и векторы d_ρ , используемые в $\hat{x}_1, \dots, \hat{x}_{m_1-1}$ для ортогонализации, хранятся так, что запоминаются лишь элементы, отличные от нуля.

2.3. Описание процедур

Общими для всех четырех рассматриваемых процедур являются обозначения следующих формальных параметров:

a, b – концы отрезка, на котором решается задача (I.I) – (I.2) – (I.3);

m – целое положительное число, определяющее число точек x_s, в которых будут получены ответы;

n – целое положительное число – порядок системы (I.I);

ka – целое положительное число, меньшее n, – число строк матрицы ψ_a (тем самым $n - ka$ – число строк матрицы ψ_b), в предыдущем изложении это число обозначалось k_a ;

ua – массив $ua[1:ka, 1:n+1]$ – расширенная матрица граничных условий в точке a , матрица $\|\psi_a \mid g_a\|$;

ub – массив $ub[1:n-ka, 1:n+1]$ – расширенная матрица граничных условий в точке b , матрица $\|\psi_b \mid g_b\|$;

pf – идентификатор процедуры с тремя параметрами, которая должна быть описана в программе, использующей рассматриваемую процедуру; процедура $pf(x, p, f)$ вычисляет для данного x значения, заполняющие массив $p[1:n, 1:n]$ (коэффициенты уравнения (I.I)) и массив $f[1:n]$ (правые части уравнения (I.I)); n – порядок системы (I.I);

eps – положительное число, регулирующее течение процесса вычислений.

Кроме того, в списки формальных параметров рассматриваемых процедур входят следующие параметры.

2.3.I. Для методов, предложенных в [I8] и [I9]:

$m1, m2$ – целые положительные числа, употребляемые пользователем для оптимизации хода вычислений (см. 2.2.I, где эти величины обозначены соответственно m_1 и m_2);

y – массив $y[0:m,1:n]$ – массив ответов, $y[s,i]$ – значение $y_i(x_s)$;

lp – метка, на которую происходит переход в случае плохой обусловленности задачи (точнее, если при решении системы (2.3) какой-либо из главных элементов по абсолютной величине окажется меньше eps).

Имя процедуры и совокупность формальных параметров имеют вид:
для метода [I8]

$as1d(a,b,m,m1,m2,n,ka,ua,ub,y,pf,eps,lp)$;

для метода [I9]

$us1d(a,b,m,m1,m2,n,ka,ua,ub,y,pf,eps,lp)$.

2.3.2. Для метода Бахвалова:

c – дополнительная точка на отрезке $[a,b]$, эта точка задается пользователем в целях оптимизации счета;

$m1$ – целое положительное число; интегрирование на отрезке $[a,c]$ проводится с постоянным шагом $(c-a)/m1/ma$; на $[b,c]$ – с постоянным шагом $(c-b)/m1/mb$;

ma – целое положительное число, определяющее число точек x_s на отрезке $[a,c]$;

mb – целое положительное число, определяющее число точек x_s на отрезке $[b,c]$;

y – массив $y[0:ma+mb,1:n]$ – массив ответов в точках $a,\dots,c,\dots,b$; $y[j,i]=y_i(x_j)$;

$epsa$ – число, вычисляемое в результате работы процедуры и позволяющее судить о точности счета:

$$eps\,a = |Z^*(c)\,Z(c) - E| ;$$

$epsb$ – число, вычисляемое в результате работы процедуры и позволяющее судить о точности счета:

$$eps\,b = |\bar{Z}^*(c)\,\bar{Z}(c) - E| ;$$

lp – метка, на которую происходит переход в случае плохой обусловленности задачи (точнее, если при решении одной из систем линейных алгебраических уравнений все элементы очередной строки окажутся по модулю меньше eps).

Имя процедуры и совокупность формальных параметров имеют вид:

$bas1d(a,b,c,ma,mb,m1,h,ka,ua,ub,y,pf,$
$epsa,epsb,eps,lp)$.

2.3.3. Для метода Годунова:

x – массив $x[0:m]$, содержащий значения x_s ;

$m1$ – целое положительное число – число участков ортогонализации (в 2.I, 2.2 это число обозначено m_1);

$m2$ – целое положительное число, которое определяет начальный шаг интегрирования задачи Коши;

y – массив $y[0:m,1:n]$ – массив ответов; $y[j,i]=y_i(x_j)$;

eps – целое положительное число; используется для контроля точности в процедуре $urkvh$;

lp – метка, переход на которую происходит в следующих случаях:
а) ранг матрицы φ_a меньше k_a ;
б) шаг интегрирования при решении какой-либо из вспомогательных задач Коши стал меньше 10^{-18};
в) система линейных алгебраических уравнений (2.6) имеет какой-либо главный элемент по модулю меньший, чем 10^{-18}.

Имя процедуры и совокупность формальных параметров имеют вид:

$gs1d(a,b,m,m1,m2,x,n,ka,ua,ub,y,pf,eps,lp)$.

§ 3. Двухточечная краевая задача с нераспавшимися краевыми условиями

Для решения задачи (I.I), (I.4) в $LTPBVP$ использован прием Мошинского (см. [2I]); сообщение о реализации алгоритма в виде процедуры на АЛГОЛе см. в [36].

3.I. Метод решения

Введем

$$y_1(x)=\begin{Vmatrix} y(x) \\ y(a+b-x) \end{Vmatrix},\quad P_1(x)=\begin{Vmatrix} -P(x) & 0 \\ 0 & P(a+b-x) \end{Vmatrix},$$

$$Q=\|E\,|\,-E\|,\quad R=\|A\,|\,B\|,\quad f_1(x)=\begin{Vmatrix} f(x) \\ -f(a+b-x) \end{Vmatrix}.$$

Мы получим вспомогательную задачу с распавшимися условиями

$$\frac{dy_1}{dx} - P_1(x)\,y_1 = f_1 ,\quad x\in\left[a,\ \frac{a+b}{2}\right] , \tag{3.I}$$

$$Ry_1(a) = C ,\qquad Qy_1\left(\frac{a+b}{2}\right) = 0 . \tag{3.2}$$

3.2. Реализация метода

Для решения задачи (3.I) – (3.2) в $LTPBVP$ применена про-

цедура $as1d$ (см. § 2).

Имя процедуры и совокупность формальных параметров для задачи (I.I), (I.4) имеют вид:

$ms1d(a, b, m, m1, m2, n, u, y, pf, eps, lp)$.

Здесь u - массив, $u[1:n, 1:2\times n+1]$ - матрица $\|A:B:C\|$. Смысл остальных формальных параметров тот же, что и для процедуры $as1d$ (см. 2.3); m - обязательно четное (это вызвано тем, что в рассматриваемом методе отрезок $[a, \frac{a+b}{2}]$ делится на $m/2$ равных частей).

Дополнение к главе III

Матричная классическая прогонка для систем дифференциальных уравнений первого порядка

Метод классической прогонки (см. [I6,I7]) используется здесь только для случая, когда порядок системы о.д.у. четен, а число граничных условий в точке A равно числу граничных условий в точке B. Сообщение о реализации алгоритма на ФОРТРАНе см. в [3I]. Обозначения, которые здесь используются, несколько отличаются от обозначений, использованных в главе III.

I. Формулировка задачи

Рассмотрим на отрезке $[A,B]$ (возможны случаи $A<B$ и $A>B$) систему о.д.у.:

$$\begin{aligned} y_1' &= P_{11}(x)\,y_1 + P_{12}(x)\,y_2 + f_1(x), \\ y_2' &= P_{21}(x)\,y_1 + P_{22}(x)\,y_2 + f_2(x) \end{aligned} \qquad (1)$$

с граничными условиями:

$$\begin{aligned} A_1 y_1(A) + B_1 y_2(A) &= g_1, \\ A_2 y_1(B) + B_2 y_2(B) &= g_2. \end{aligned} \qquad (2)$$

Здесь N есть заданное целое положительное число, $2N$ - порядок системы; $P_{11}(x), P_{12}(x), P_{21}(x), P_{22}(x)$ - заданные при $x\in[A,B]$ квадратные матрицы порядка N; $f_1(x), f_2(x)$ - заданные при $x\in[A,B]$ столбцы высоты N; A_i, B_i $(i=1,2)$ - заданные квадратные матрицы порядка N; g_1, g_2 - заданные столбцы высоты N. Далее предполагаем, что функции $P_{ij}(x)$ и $f_j(x)$ $(i=1,2; j=1,2)$ непрерывны на $[A,B]$.

2. Метод решения

По всему отрезку $[A,B]$ переносим соотношение

$$y_1 = \alpha(x)\,y_2 + \beta(x),$$

решая соответствующие задачи Коши для уравнений классической прогонки:

$$\begin{aligned} \alpha' &= P_{11}\alpha - \alpha P_{22} - \alpha P_{21}\alpha + P_{12}, \\ \beta' &= P_{11}\beta - \alpha P_{21}\beta - \alpha f_2 + f_1. \end{aligned} \qquad (3)$$

Здесь $\alpha(x)$ - квадратная матрица порядка N, $\beta(x)$ - столбец высоты N. От A к B решаем систему (3) при начальных условиях:

$$\alpha(A) = -A_1^{-1}B_1, \quad \beta(A) = A_1^{-1}g_1.$$

При этом запоминаем значения $\alpha_A(x)$ и $\beta_A(x)$ в тех точках x_j, $j=0,1,\dots,M$, отрезка $[A,B]$, где мы хотим получить ответы.

От B к A интегрируем уравнения (3) с условиями:

$$\alpha(B) = -A_2^{-1}B_2, \quad \beta(B) = A_2^{-1}g_2,$$ вычисляя

$$\alpha_B(x_j),\ \beta_B(x_j),\quad j=M, M-1, \dots, 0.$$

Искомое решение в заданных точках x_j, $j=0,1,\dots,M$, получается из системы линейных алгебраических уравнений

$$\begin{aligned} y_1(x_j) &= \alpha_A(x_j)\,y_2(x_j) + \beta_A(x_j), \\ y_1(x_j) &= \alpha_B(x_j)\,y_2(x_j) + \beta_B(x_j), \quad j=0,1,\dots,M. \end{aligned} \qquad (4)$$

3. Реализация метода, описание программы

Решение задачи (I) - (2) определяется в точках $x_j = A + j\times(B-A)/M$, $j=0,1,\dots,M$. Численное решение вспомогательных задач Коши осуществляется методом Рунге-Кутта четвертого порядка точности с автоматическим выбором шага интегрирования с абсолютной точностью $EPS/(M\times M1)$ на каждом из отрезков $[x_j, x_{j+1}]$. Целое положительное число $M1$ задает начальный шаг интегрирования H,

$$H = (B-A)/(M\times M1).$$

Решение системы линейных алгебраических уравнений (4) и обращение матриц A_1 и A_2 осуществляются методом Гаусса с выбором главных элементов по столбцам.

Заголовок подпрограммы $CLS1D$ имеет вид:

SUBROUTINE CLS1D (A,B,M,M1,M2,N,Y,EPS,SIGM).

Формальные параметры имеют следующий смысл:

A,B – концы интервала интегрирования;

M – целое положительное число, задающее количество точек x_s, в которых будут получены ответы;
$x_s = A + S \times (B-A)/M$, $s = 0,1,\ldots,M$;

M1 – целое положительное число, определяющее начальный шаг интегрирования H в методе Рунге-Кутта, $H=(B-A)/(M \times M1)$;

M2 – целое положительное число, $M2 = M+1$;

N – целое положительное число; порядок систему о.д.у. равен $2 \times N$;

Y – массив размерности $(2, N, M2)$, в котором записывается решение исходной задачи – два столбца Y_1 и Y_2 высоты N со значениями, вычисленными в заданных точках x_s, $s = 0,1,\ldots,M$.

EPS – точность в методе Рунге-Кутта: задача Коши решается с абсолютной точностью $EPS/(M1 \times M)$;

SIGM – логическая переменная. При обращении к подпрограмме переменная SIGM должна иметь значение TRUE. В случае отказа в работе подпрограммы SIGM принимают значение FALSE.

С целью сокращения времени счета, экономии памяти машины и уменьшения числа формальных параметров в подпрограмме CLS1D использованы следующие COMMON-блоки:

COMMON /FUN/ C1 (N, N + 1)
COMMON /SOLU/ C2 (N, N + 1, M)
COMMON /RIP/ C3 (N, 5N + 2)
COMMON /COIN/ C4 (N, 4N + 2)
COMMON /RK/ C5 (4N, N + 1)
COMMON /RES/ C6 (2 N)
COMMON /RES1/ C7 (N, N + 1)
COMMON /COIN1/ C8 (N, N)
COMMON /SLE/ C9 (N)
COMMON /BOUN/ C10 (4N)

Эти COMMON-блоки должны содержаться в вызываемой программе пользователя. Размерности массивов $C_1, C_2, \ldots, C_{10}$, а также массива ответов Y в вызывающей программе должны быть описаны в соответствии со значениями параметров M, N, N1. Вместо этих параметров в описаниях должны быть подставлены их численные значения.

Для задания матриц, определяющих граничные условия, пользователь должен составить подпрограмму CONDIB, обращение к которой имеет вид:

SUBROUTINE CONDIB
COMMON /COIN/ C4 (1)
COMMON /KCOIN/ N1, N2, JA1, JA2, JB1, JB2, J61, J62.

Здесь целые параметры JA1, JA2, JB1, JB2, J61, J62 – это метки, определяющие положение элементов граничных матриц A_1, A_2, B_1, B_2, g_1, g_2 в массиве C4. Например, элементы матрицы A_1 располагаются в массиве C4 по столбцам, начиная с элемента под номером JA1 + 1. Размерность массива C4 указывается в вызывающей программе и равна $N(4N+2)$.

Для задания матриц, определяющих систему о.д.у., служит подпрограмма пользователя RIPART, обращение к которой имеет вид:

SUBROUTINE RIPART (X)
COMMON /RIP/ C3 (1)
COMMON /KPIP/ N, JP11, JP12, JP21, JP22, JF1, JF2.

Здесь целые параметры JP11, JP12, JP21, J22, JF1, JF2 – это метки, определяющие положение элементов матриц $P_{11}, P_{12}, P_{21}, P_{22}, F_1, F_2$ в массиве C3 так же, как это было сделано в подпрограмме CONDIB. Размерность массива C3 указывается в вызывающей подпрограмме и равна $N(5N+2)$.

В подпрограмме CLS1D в случае, если для рассматриваемой задачи метод классической прогонки не пригоден, предусмотрено прерывание процесса решения с указанием причин отказа. Отказ происходит в следующих случаях:

1) если шаг интегрирования вспомогательных задач Коши стал меньше чем $10^{-10}|B-A|$, необходимая точность решения не достигнута;
2) если при обращении матриц A_1, A_2 или при решении разрешающих систем алгебраических уравнений (4) главный элемент какого-либо столбца оказался меньше $EPS/(M \times M1)$.

Глава IУ. Заключительные замечания и примеры

В предыдущих главах мы, в частности, отмечали, при каких предположениях относительно исходной краевой задачи гарантируется ее решение тем или иным методом. Здесь мы приведем примеры задач, которые не решаются тем или иным из описанных методов или ни одним из них. Причины, по которым это происходит и которые мы укажем, приводят в соответствующих процедурах к выходу на метки lm или lp. Важную роль

при этом играет понятие корректности решаемой краевой задачи. Вопрос о том, какие краевые задачи устойчивы относительно малых изменений входящих в них величин, очень труден, некоторые простые соображения по этому вопросу см. в [8 , гл. IX].

§ I. Примеры краевых задач, не решаемых классическими вариантами прогонки

I.I. Краевая задача

$$y'' + y = 0, \quad x \in [a,b], \qquad y(a) = 0, \quad y(b) = 1, \tag{4.1}$$

где $(b-a) \neq k\pi$, $k = 0, \pm 1, \dots$, имеет единственное решение

$$y(x) = \frac{\sin(x-a)}{\sin(b-a)} .$$

Для задачи (4.1), если $|b-a| \geq \pi/2$, не выполнены достаточные условия устойчивости метода классической прогонки: при решении этой задачи с помощью одной из процедур *clod*, *stclod*, *cls 1d* или *cls 2d* происходит выход на метку *lm* – "неподходящий метод". Эта задача может быть решена с помощью любой из описанных выше процедур, реализующей "неклассический" метод прогонки (например, *aod*).

I.2. Краевая задача

$$y'' + y = -3\sin 2x, \quad 0 \leq x \leq \pi/2 , \tag{4.2}$$

$$y(0) = y(\pi/2), \quad y'(0) = y'(\pi/2) \tag{4.3}$$

с граничными условиями периодичности имеет единственное решение

$$y(x) = -2(\cos x + \sin x) + \sin 2x .$$

Однако при использовании для ее решения процедуры *cyclod* происходит выход на метку *lm* – "неподходящий метод": для уравнения (4.2) не выполнены достаточные условия устойчивости метода классической прогонки (этот метод используется в процедуре *cyclod* для решения вспомогательных краевых задач с разделенными краевыми условиями). Для решения задачи (4.2), (4.3) можно воспользоваться приемом [2I] для разделения связанных граничных условий (см. п.I.2 главы I), и для решения полученной самосопряженной краевой задачи для системы двух уравнений второго порядка с распавшимися краевыми условиями использовать процедуру *bts2d* . Полученная задача может быть также решена с помощью любой другой процедуры, реализующей какой-либо вариант ортогональной прогонки и предназначенной для решения систем уравнений. Кроме того, можно свести задачу (4.2), (4.3) к задаче для системы из двух уравнений первого порядка с неразделенными краевыми условиями и использовать для ее решения процедуру *ms1d* .

§ 2. Примеры плохо обусловленных краевых задач

2.I. Краевая задача

$$y'' = x, \quad -1 \leq x \leq 1, \qquad y'(-1) = y'(1) = 0 \tag{4.4}$$

является плохо обусловленной, ее решение

$$y(x) = \frac{x^3}{6} - \frac{x}{2} + c$$

определено с точностью до постоянного слагаемого -нетривиального решения соответствующей однородной задачи. При решении задачи (4.4) с помощью любой из описанных процедур, предназначенной для решения задачи с разделенными краевыми условиями, происходит выход на метку *lp* – "плохо обусловленная задача".

2.2. Краевая задача

$$y'' + y = 6, \quad 0 \leq x \leq 2\pi , \qquad y(0) = y(2\pi), \quad y'(0) = y'(2\pi) \tag{4.5}$$

с граничными условиями периодичности является плохо обусловленной при любом 6 , так как ее решение

$$y(x) = c_1 \sin x + c_2 \cos x + 6$$

определено с точностью до слагаемого - собственной функции однородной задачи. При использовании для решения задачи (4.5) процедуры *cyclod* выход из процедуры, однако, происходит раньше на метку *lm* – "неподходящий метод", так как развивается численная неустойчивость при решении уравнений классической прогонки.

Если привести задачу (4.5) к эквивалентной задаче для системы первого порядка и воспользоваться процедурой *ms1d* . , то будет осуществляться выход на метку *lp* – "плохо обусловленная задача".

2.3. Рассмотрим краевую задачу вида

$$y''+2\frac{\varphi'(x)}{\varphi(x)}y'+\left(\frac{\varphi''(x)}{\varphi(x)}-1\right)y=\frac{1}{\varphi(x)},\quad -a\le x\le a, \tag{4.6}$$

$$y'(-a)+\left[\frac{\varphi'(-a)}{\varphi(-a)}+1\right]y(-a)=-\frac{1}{\varphi(-a)}, \tag{4.7}$$

$$y'(a)+\left[\frac{\varphi'(a)}{\varphi(a)}-1\right]y(a)=\frac{1}{\varphi(a)}, \tag{4.8}$$

где $\varphi(x)=2+th\,x$.

Эта задача имеет единственное решение $y(x)=-\frac{1}{\varphi(x)}$. Однако для достаточно больших a задача (4.6) - (4.8) является плохо обусловленной, так как малые изменения правой части уравнения (4.6) или граничных условий (4.7), (4.8) приводят к сильным изменениям в решении. В самом деле, решение возмущенного уравнения

$$y_\varepsilon''+2\frac{\varphi'}{\varphi}y_\varepsilon'+\left(\frac{\varphi''}{\varphi}-1\right)y_\varepsilon=\frac{1+\varepsilon}{\varphi},\quad -a\le x\le a,$$

где ε мало, удовлетворяющее тем же граничным условиям (4.7), (4.8), имеет вид

$$y_\varepsilon(x)=-\frac{1}{\varphi(x)}\left[(1+\varepsilon)-\varepsilon e^{a}ch\,x\right].$$

При больших a это решение сильно отличается от решения невозмущенной задачи.

При решении задачи (4.6) - (4.8) каким-либо методом прогонки будут наблюдаться следующие явления. Граничное условие (4.7), соответствующее выделению однопараметрического семейства решений (4.6), экспоненциально возрастающих справа налево при больших a , с помощью уравнений прогонки неустойчиво переносится слева направо. Во всех процедурах будет наблюдаться дробление шага при интегрировании уравнений прогонки и либо выход на метку lm (в тех процедурах, где она предусмотрена), так как она связана с ограничением на величину шага интегрирования, либо выход на переполнение AY вследствие быстрого накопления вычислительной погрешности. То же относится к переносу правого граничного условия (4.8), соответствующего выделению однопараметрического семейства решений (4.6), экспоненциально возрастающих слева направо при больших a .

Отметим, что краевая задача (4.6) - (4.8) получена из более простой плохо обусловленной краевой задачи

$$z''-z=1,\quad -a\le x\le a, \tag{4.9}$$

$$z'(-a)=-z(-a)-1, \tag{4.10}$$

$$z'(a)=z(a)+1 \tag{4.11}$$

заменой $y(x)=z(x)/\varphi(x)$. В исходном виде задача (4.9) - (4.11) является нетипичным примером плохо обусловленной краевой задачи: она имеет единственное решение $z(x)\equiv -1$; в уравнениях прогонки отыскиваются решения типа константы, и накопления вычислительной погрешности не происходит, так что, например, с помощью процедур $clod$ или aod задача (4.9) - (4.11) успешно решается несмотря на ее некорректность.

2.4. Приведем примеры более сложных плохо обусловленных краевых задач.

Основным отладочным примером для процедур $as1d$, $bas1d$, $us1d$, $gs1d$ послужила заимствованная из [50] краевая задача вида

$$y'-x\,Ay=-\frac{x}{x+1}\,Aq-\frac{1}{(x+1)^2}q,\quad x\in[a,b], \tag{4.12}$$

$$\psi_a\,y(a)=\frac{1}{1+a}\,\psi_a\,q, \tag{4.13}$$

$$\psi_b\,y(b)=\frac{1}{1+b}\,\psi_b\,q. \tag{4.14}$$

Здесь A - постоянная квадратная матрица порядка n , q - постоянный n -столбец, ψ_a и ψ_b - постоянные матрицы размера $k\times n$ и $(n-k)\times n$ соответственно. Функция $y(x)=\frac{1}{1+x}q$ - решение задачи (4.12) - (4.14) при любых A, q, ψ_a, ψ_b .

Пусть матрица A имеет вид

$$A=\begin{Vmatrix} -2 & 2 & 1\\ 0 & 2 & 2\\ -2 & 1 & -1\end{Vmatrix}.$$

Ее собственные значения суть $\lambda_1=-2$, $\lambda_2=2$, $\lambda_3=-1$. Собственным значениям $\lambda_1, \lambda_2, \lambda_3$ отвечают следующие собственные векторы матрицы A^* :

$$f_1=\begin{Vmatrix}2\\-1\\0\end{Vmatrix}\qquad f_2=\begin{Vmatrix}-2\\7\\4\end{Vmatrix}\qquad f_3=\begin{Vmatrix}-2\\1\\1\end{Vmatrix}.$$

Зафиксируем $a=0$, $b=10$ и какое-либо q , например $q=\begin{Vmatrix}2\\-1\\1\end{Vmatrix}$.

1) Положим

$$\psi_\alpha = \begin{Vmatrix} 1 & 0 & 1 \\ 2 & 3 & 4 \end{Vmatrix}, \quad \psi_\beta = \|2 \quad -1 \quad 0\| .$$

Так как $\psi_\beta = f_1^*$, то задача (4.12) – (4.14) плохо обусловлена. При ее решении каким-либо методом прогонки будут наблюдаться следующие явления. Пусть мы сначала переносим граничное условие (4.14) справа налево. Так как это граничное условие соответствует выделению решений (4.12), ведущих себя как e^{x^2}, то граничное условие (4.14) при больших β неустойчиво переносится справа налево. Будет наблюдаться дробление шага при интегрировании уравнений прогонки или выход на переполнение AY вследствие быстрого накопления вычислительной погрешности.

Если сначала переносить граничное условие (4.13) слева направо, то при большом β строки $\psi_\alpha(\beta)$ будут порождать подпространство, почти содержащее вектор-строку ψ_β, что приведет в процедурах к выходу на метку lp – "плохо обусловленная задача".

2) Положим теперь

$$\psi_\alpha = \begin{Vmatrix} -2 & -1 & 0 \\ 0 & 8 & 4 \end{Vmatrix}, \quad \psi_\beta = \|1 \quad 0 \quad 1\| .$$

Строки ψ_α порождают подпространство, содержащее вектор-строку f_2^*. Задача (4.12) – (4.14) снова является плохо обусловленной. Граничное условие (4.13) будет неустойчиво переноситься слева направо. Если же сначала осуществить перенос граничного условия (4.14) справа налево, то в точке α будет осуществляться выход на метку lp.

§ 3. Примеры краевых задач, не имеющих решения

3.1. Краевая задача

$$\begin{aligned} &y'' + y = 0, && 0 \le x \le \pi, \\ &y(0) = 0, && y(\pi) = \sigma \end{aligned} \tag{4.15}$$

не имеет решения при любом $\sigma \neq 0$, а соответствующая однородная задача имеет нетривиальное решение. При использовании для решения задачи (4.15) какой-либо из процедур *clod, stclod, cls2d* или *cls1d*, реализующих методы классической прогонки, происходит выход на метку lm – "неподходящий метод", так как развивается численная неустойчивость при решении уравнений классической прогонки. При использовании для решения задачи (4.15) какой-либо из процедур, реализующей один из вариантов ортогональной прогонки, происходит выход на

метку lp – "некорректная задача".

3.2. Краевая задача

$$\begin{aligned} &y'' = \sigma, && 0 \le x \le 2\pi, \\ &y(0) = y(2\pi), && y'(0) = y'(2\pi) \end{aligned} \tag{4.16}$$

с граничными условиями периодичности не имеет решения при любом $\sigma \neq 0$, а соответствующая однородная задача имеет нетривиальное решение. При использовании для решения задачи (4.16) процедуры *cyclod* происходит выход на метку lp – "некорректная задача".

Часть II. Разностная прогонка

Введение

В настоящей главе рассмотрены алгоритмы некоторых видов классических прогонок, предназначенных для решения разностных краевых задач. Алгоритмы входят в состав пакета *LTPBVP* и реализованы на алгоритмических языках АЛГОЛ и ФОРТРАН. В пакете представлены следующие варианты разностных прогонок:

- алгоритм трехточечной скалярной прогонки (см. § 1);
- алгоритм потоковой прогонки (см. § 2);
- алгоритм трехточечной скалярной прогонки с условием периодичности (см. § 3);
- алгоритм многоточечной скалярной прогонки (см. § 4);
- алгоритм трехточечной матричной прогонки (см. § 5).

Каждый алгоритм реализован в виде отдельной процедуры (подпрограммы). В пакет не включены алгоритмы общей многоточечной матричной прогонки (см. [24]) и некоторые варианты ортогональных разностных прогонок (см. [42, 43]). Приводится достаточно подробное описание алгоритмов и некоторые сведения, касающиеся их программной реализации. Более подробную информацию о программах можно найти в [37–41].

§ 1. Алгоритм решения разностной краевой задачи методом трехточечной скалярной прогонки

В этом параграфе представлен алгоритм решения систем линейных алгебраических уравнений, матрицы которых имеют трехдиагональный вид. Такие системы возникают, в частности, при решении одного линейного обыкновенного дифференциального уравнения второго порядка с заданными граничными условиями методом конечных разностей. Метод трехточечной скалярной прогонки рассматривался в [5, 22]. Программа, реали-

зующая алгоритм, написана на языке ФОРТРАН.

1. Описание алгоритма

Задана система линейных алгебраических уравнений

$$\begin{aligned} &y_0 = f_0, \\ &a_i y_{i-1} - c_i y_i + b_i y_{i+1} = f_i, \quad 1 \le i \le M-1, \\ &y_M = f_M. \end{aligned} \tag{I.1}$$

Здесь a_i, c_i, b_i, f_0, f_M – заданные числа.

Введем следующие прогоночные соотношения

$$\begin{aligned} &y_i = \ell_i y_{i+1} + k_i, \qquad i = 1, \dots, M-1, \\ &\ell_1 = c_1^{-1} b_1, \qquad k_1 = c_1^{-1}(a_1 f_0 - f_1), \\ &\ell_i = (c_i - a_i \ell_{i-1})^{-1} b_i, \quad k_i = (c_i - a_i \ell_{i-1})^{-1}(a_i k_{i-1} - f_i). \end{aligned} \tag{I.2}$$

где

Предполагая, что

$$|\ell_1| > eps, \quad |c_i - a_i \ell_{i-1}| > eps, \quad i = 1, \dots, M-2,$$

где eps – заданное малое положительное число. В противном случае считается, что метод с данной точностью не может быть применен для решения задачи. Предполагается также, что

$$|c_{M-1} - a_{M-1} \ell_{M-2}| > eps.$$

В противном случае система (I.1) с заданной точностью считается плохо обусловленной системой. При $i = M-1$ имеем

$$y_{M-1} = \ell_{M-1} y_M + k_{M-1}. \tag{I.3}$$

Используя граничное условие $y_M = f_M$ из (I.3) находим y_{M-1}. После этого из (I.2) определяем значения y_i, $i = M-2, \dots, 1$. Достаточными условиями устойчивости рассматриваемого метода являются условия

$$a_i > 0, \quad b_i > 0, \quad c_i \ge a_i + b_i.$$

2. Программная реализация

Обращение к подпрограмме решения системы (I.1) имеет вид

CALL OF (M1, M11, Y, A, C, B, F, EPS).

Здесь:

$M1 = N+1$ – целое число;

$M11 = N-1$ – целое число;

Y – массив, в котором помещается искомое решение системы (I.1) и известные решения на границе;

A, C, B – массивы, в которые пользователь помещает значения коэффициентов системы (I.1) a_i, c_i, b_i, $i = 1, \dots, M-1$, соответственно;

F – массив, в который пользователь помещает на первое место значение f_0, ячейки со 2-й по M-ю отводятся для f_i, $i = 1, \dots, M-1$, а в ячейку $M+1$ помещается значение f_M;

EPS – малое положительное число, задаваемое пользователем.

§ 2. Алгоритм решения разностной краевой задачи методом потоковой прогонки

В данном параграфе представлен алгоритм решения систем линейных алгебраических уравнений, возникающих при разностной аппроксимации дифференциальных уравнений с сильно меняющимися коэффициентами (см. [9, 23]). Такие системы возникают, например, при решении краевой задачи для уравнения $\frac{d}{dx}\left(\frac{1}{\sigma}\frac{dy}{dx}\right) - qy = f$, где коэффициент $1/\sigma$ может сильно меняться. Предлагается ввести с помощью замены переменной вспомогательную функцию W по формуле $W = -\frac{1}{\sigma}\frac{dy}{dx}$ и решать соответствующую краевую задачу для системы уравнений

$$\begin{aligned} &-\frac{dW}{dx} - qy = f, \\ &\frac{dy}{dx} + \sigma W = 0. \end{aligned}$$

Программа, реализующая соответствующий разностный алгоритм, написана на языке АЛГОЛ-60.

1. Описание алгоритма

Задана система линейных алгебраических уравнений

$$\alpha_1 y_0 = \alpha_2 W_0 + \alpha_3, \tag{2.1}$$

$$a_i W_i - c_i W_{i+1} + b_i y_i = f_i, \tag{2.2}$$

$$W_{i+1} = \frac{y_i - y_{i+1}}{g_{i+1}}, \qquad i = 0, \dots, m-1, \tag{2.3}$$

$$\beta_1 y_m = \beta_2 W_m + \beta_3. \tag{2.4}$$

Если $\alpha_1 \neq 0$, то ищем такие $\tilde{\ell}_i$ $\tilde{k}_i$, чтобы выполнялись соотношения

$$W_i = \tilde{\ell}_i W_{i+1} + \tilde{k}_i, \tag{2.5}$$

$$y_{i+1} = \ell_{i+1} W_{i+1} + k_{i+1}. \tag{2.6}$$

Для прогоночных коэффициентов $\ell_i, k_i, \tilde{\ell}_i, \tilde{k}_i$ получим следующие формулы:

$$\ell_0 = \frac{\alpha_2}{\alpha_1}, \quad k_0 = \frac{\alpha_3}{\alpha_1}, \quad \tilde{\ell}_0 = \frac{c_0}{a_0 + b_0 \ell_0}, \quad \tilde{k}_0 = \frac{f_0 - b_0 k_0}{a_0 + b_0 \ell_0},$$

$$\tilde{\ell}_i = \frac{c_i}{a_i + b_i \ell_i}, \qquad \tilde{k}_i = \frac{f_i - b_i k_i}{a_i + b_i \ell_i}, \tag{2.7}$$

$$\ell_{i+1} = \frac{\ell_i c_i}{a_i + b_i \ell_i} - g_{i+1}, \quad k_{i+1} = \frac{\ell_i f_i + k_i a_i}{a_i + b_i \ell_i}.$$

Далее, $w_m = \dfrac{\beta_1 k_m - \beta_3}{\beta_2 - \beta_1 \ell_m}$,

$$y_m = \frac{\beta_2}{\beta_1} w_m + \frac{\beta_3}{\beta_1}, \quad \text{если } \beta_1 \neq 0, \tag{2.8}$$
$$y_m = \ell_m w_m + k_m, \quad \text{если } \beta_1 = 0.$$

Затем пользуемся формулами (2.5) и (2.3) для нахождения искомого решения.

В случае $\alpha_2 \neq 0$ ищем прогоночные коэффициенты $\ell_i, k_i, \ell_i, \tilde{k}_i$, чтобы выполнялись соотношения

$$w_{i+1} = \ell_{i+1} y_{i+1} + k_{i+1}, \tag{2.9}$$
$$y_i = \tilde{\ell}_i w_{i+1} + \tilde{k}_i. \tag{2.10}$$

Имеем $\ell_0 = \dfrac{\alpha_1}{\alpha_2}$, $k_0 = -\dfrac{\alpha_3}{\alpha_2}$, $\tilde{\ell}_i = \dfrac{c_i}{a_i \ell_i + b_i}$, $\tilde{k}_i = \dfrac{f_i - a_i k_i}{a_i \ell_i + b_i}$,

$$\ell_{i+1} = \frac{a_i \ell_i + b_i}{c_i - g_{i+1}(a_i \ell_i + b_i)}, \quad k_{i+1} = \frac{a_i k_i - f_i}{c_i - g_{i+1}(a_i \ell_i + b_i)}, \tag{2.11}$$

$$y_m = \frac{\beta_3 + \beta_2 k_m}{\beta_1 - \beta_2 \ell_m},$$

$$w_m = (\beta_1/\beta_2) y_m - \beta_3/\beta_2, \quad \text{если } \beta_2 \neq 0, \tag{2.12}$$
$$w_m = \ell_m y_m + k_m, \quad \text{если } \beta_2 = 0.$$

При реализации метода предполагается, что выполняются условия:

$$|\alpha_i + b_i \ell_i| > eps \quad \text{в случае } \alpha_1 \neq 0, \tag{2.13}$$
$$|a_i \ell_i + b_i| > eps \quad \text{в случае } \alpha_2 \neq 0, \text{ где}$$

eps – заданное малое положительное число. В противном случае считается, что метод с данной точностью не может быть применен для решения задачи. Предполагается также, что

$$|\beta_2 - \beta_1 \ell_m| > eps \quad \text{в случае } \alpha_1 \neq 0, \tag{2.14}$$
$$|\beta_1 - \beta_2 \ell_m| > eps \quad \text{в случае } \alpha_2 \neq 0.$$

В противном случае система (2.1 – 2.4) с заданной точностью считается плохо обусловленной системой. Процедура решения описанной задачи предусматривает граничные условия только следующего вида:

$$y_0 = \alpha_2 w_0 + \alpha_3, \tag{2.15}$$
$$y_m = \beta_2 w_m + \beta_3, \tag{2.16}$$
$$w_0 + \alpha_3 = 0, \tag{2.17}$$
$$w_m + \beta_3 = 0. \tag{2.18}$$

Достаточными условиями устойчивости метода для задач с граничными условиями (2.15); (2.16) являются условия

$$a_i \geq c_i > 0, \qquad b_i < 0,$$

для задачи с условиями (2.17), (2.18) – условия

$$c_i \geq a_i > 0, \qquad b_i < 0.$$

2. Программная реализация

Имя процедуры и список формальных параметров имеет вид

stc_lof(m, ua, ub, y, w, a, c, b, f, g, eps, lm, lp).

Здесь

m – число уравнений в (2.2). Общее число уравнений системы, таким образом, равно $2m$;

элементы массивов $ua[1:3]$ и $ub[1:3]$ суть соответственно

$ua[1] = \alpha_1$, $ub[1] = \beta_1$,
$ua[2] = \alpha_2$, $ub[2] = \beta_2$,
$ua[3] = \alpha_3$, $ub[3] = \beta_3$;

$y[0:m]$, $w[0:m]$ – массивы ответов;

$a, c, b, f[0:m-1]$, $g[1:m]$ – массивы коэффициентов a_i, c_i, b_i, f_i, g_i системы (2.2), (2.3);

eps – малая положительная величина, задаваемая пользователем. Если $|\alpha_1| < eps$ и $|\beta_1| < eps$, то решается задача с граничными условиями (2.15), (2.16). В противном случае решается задача с условиями (2.17), (2.18);

lm – метка, на которую выходит программа в случае, когда не выполняются условия (2.13);

lp – метка, на которую выходит программа в случае, когда не выполняются условия (2.14).

§ 3. Алгоритм решения разностной краевой задачи с условием периодичности методом трехточечной скалярной прогонки

Алгоритм предназначен для решения системы линейных алгебраических уравнений, матрица которых имеет следующий вид:

$$A = \begin{Vmatrix} -c_1 & b_1 & 0 & \dots & 0 & 0 & a_1 \\ a_2 & -c_2 & b_2 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & & a_{M-1} & -c_{M-1} & b_{M-1} \\ b_M & 0 & 0 & & 0 & a_M & -c_M \end{Vmatrix} .$$

Такие системы возникают, например, при разностной аппроксимации краевой задачи для одного линейного обыкновенного дифференциального уравнения второго порядка с условием периодичности. Для решения такой задачи может быть применен метод разностной прогонки [10]. Программа, реализующая алгоритм, написана на языке АЛГОЛ-60.

1. Описание алгоритма

Задана система линейных алгебраических уравнений

$$\begin{aligned} -c_1 y_1 + b_1 y_2 + a_1 y_M &= f_1 , \\ a_i y_{i-1} - c_i y_i + b_i y_{i+1} &= f_i , \quad 2 \le i \le M-1 , \\ b_M y_1 + a_M y_{M-1} - c_M y_M &= f_M \end{aligned} \qquad (3.1)$$

Решение системы (3.1) находится по следующим формулам:

$$y_M = \frac{\beta_M + \ell_M p_1}{1 - \ell_M q_1 - \gamma_M} ,$$

$$y_i = p_i + q_i y_{i+1} , \qquad i = M-1, \dots, 1 .$$

Коэффициенты p_i и q_i находятся из рекуррентных соотношений

$$p_i = \ell_i p_{i+1} + \beta_i ,$$

$$q_i = \ell_i q_{i+1} + \gamma_i , \quad i = M-2, \dots, 1 ,$$

$$p_{M-1} = \beta_{M-1} ,$$

$$q_{M-1} = \ell_{M-1} + \gamma_{M-1} , \quad \text{где}$$

$$\ell_1 = \frac{b_1}{c_1} , \quad \beta_1 = -\frac{f_1}{c_1} , \quad \gamma_1 = \frac{a_1}{c_1} ,$$

$$\ell_i = \frac{b_i}{c_i - a_i \ell_{i-1}} , \quad \beta_i = \frac{a_i \beta_{i-1} - f_i}{c_i - a_i \ell_{i-1}} , \quad \gamma_i = \frac{a_i \gamma_{i-1}}{c_i - a_i \ell_{i-1}} , \quad i = 2, \dots, M .$$

Предполагается, что

$$|c_1| > eps \quad \text{и} \quad |c_i - a_i \ell_{i-1}| > eps , \quad i = 2, \dots, M , \qquad (3.2)$$

где eps – заданное малое положительное число. В противном случае считается, что метод с данной точностью не может быть применен для решения задачи. Предполагается также, что

$$|1 - \ell_M q_1 - \gamma_M| > eps . \qquad (3.3)$$

В противном случае система (3.1) с заданной точностью считается плохо обусловленной системой. Достаточными условиями устойчивости рассматриваемого метода являются условия

$$a_i > 0, \quad b_i > 0, \quad c_i > a_i + b_i .$$

2. Программная реализация

Имя процедуры и совокупность формальных параметров имеют вид:

$$pclof\ (m, y, a, c, b, f, eps, lm, lp) .$$

Здесь:

m – порядок системы;

массивы $a[1:m]$, $c[1:m]$, $b[1:m]$ – массивы коэффициентов системы (3.1);

$f[1:m]$ – массив правых частей системы (3.1);

$y[1:m]$ – массив ответов;

eps – малая положительная величина, задаваемая пользователем;

lm – метка, на которую выходит программа в тех случаях, когда не выполняются условия (3.4);

lp – метка, на которую выходит программа в случае, когда не выполняется условие (3.5).

§ 4. Алгоритм решения разностной краевой задачи методом многоточечной скалярной прогонки

Алгоритм предназначен для решения систем линейных алгебраических уравнений, матрицы которых имеют $(n+1)$ – диагональный вид (см. [24]). Такие системы возникают, в частности, при решении одного линейного обыкновенного дифференциального уравнения n -го порядка с заданными граничными условиями методом конечных разностей. Программа, реали-

зующая алгоритм, написана на языке АЛГОЛ-60.

I. Описание алгоритма

Задана система линейных алгебраических уравнений

$$y_i + a^{(0)}_{i,1} y_{i+1} + \ldots + a^{(0)}_{i,n} y_{i+n} = f^{(0)}_i, \qquad i = 0, \ldots, M-n. \tag{4.I}$$

Здесь $f^{(0)}_i, a^{(0)}_{i,1}, \ldots, a^{(0)}_{i,n}$ – заданные числа. Для выделения единственного решения рассмотрим еще две группы уравнений, которые соответствуют левым и правым граничным условиям. Имеем

$$y_0 + a^{(i)}_{0,1} + \cdots + a^{(i)}_{0,n-i} y_{n-1} = f^{(i)}_0, \qquad i = 1, \ldots, I, \tag{4.2}$$

$f^{(i)}_0, a^{(i)}_{0,1} \ldots, a^{(i)}_{0,n-i}$ – заданные числа,

$$a^{(1)}_{i,1} y_{M-n+1} + a^{(1)}_{i,2} y_{M-n+2} + \ldots + a^{(1)}_{i,n} y_M = f^{(1)}_i, \quad i = 1, \ldots, n-I, \tag{4.3}$$

$f^{(1)}_i, a^{(1)}_{i,1}, \ldots, a^{(1)}_{i,n}$ – заданные числа.

Решение системы ищется в виде:

$$y_i + a^{(1)}_{i,1} y_{i+1} + \cdots + a^{(1)}_{i,n-1} y_{i+n-1} = f^{(1)}_i. \tag{4.4}$$

Используя систему (4.I), переносим соотношение (4.4) из точки с номером i в точку с номером $i+1$.

$$y_{i+1} + a^{(1)}_{i+1,1} y_{i+2} + \ldots + a^{(1)}_{i+1,n-1} y_{i+n} = f^{(1)}_{i+1}$$

, где

$$l^{(1)}_i = (a^{(0)}_{i,1} - a^{(1)}_{i,1})^{-1},$$

$$a^{(1)}_{i+1,1} = l^{(1)}_i (a^{(0)}_{i,2} - a^{(1)}_{i,2}),$$

$$\overline{a^{(1)}_{i+1,n-2} = l^{(1)}_i (a^{(0)}_{i,n-1} - a^{(1)}_{i,n-1})},$$

$$a^{(1)}_{i+1,n-1} = l^{(1)}_i a^{(0)}_{i,n},$$

$$f^{(1)}_{i+1} = l^{(1)}_i (f^{(0)}_i - f^{(1)}_i).$$

Предполагая, что

$$|a^{(0)}_{i,1} - a^{(1)}_{i,1}| > eps, \tag{4.5}$$

где eps – заданное малое положительное число. В противном случае считается, что метод с данной точностью не может быть применен для решения задачи. Продолжая первую прямую прогонку до точки $M-n+1$

используем соотношение

$$y_{M-n+1} + a^{(1)}_{M-n+1,1} y_{M-n+2} + \ldots + a^{(1)}_{M,1} y_M = f^{(1)}_{M-n+1}$$

для понижения порядка правых граничных условий. Они примут вид

$$a^{(2)}_{i,2} y_{M-n+2} + a^{(2)}_{i,3} y_{M-n+3} + \cdots + a^{(2)}_{i,n} y_M = f^{(2)}_i, \qquad i = 1, \ldots, n-I.$$

Здесь

$$a^{(2)}_{i,k} = a^{(1)}_{M-n-1+k} - \frac{a^{(1)}_{i\,k}}{a^{(1)}_{i,1}}, \qquad k = 2, \ldots, n,$$

$$f^{(2)}_i = f^{(1)}_{M-n+1} - f^{(1)}_i / a^{(1)}_{i,1}.$$

После переноса всех левых граничных условий правые граничные условия примут вид

$$a^{(I+1)}_{i,I+1} y_{M-n+I+1} + \ldots + a^{(I+1)}_{i,n} y_M = f^{(I+1)}_i, \qquad i = 1, \ldots, n-I.$$

Обозначим величину определителя этой системы через det. Предполагается, что

$$|det| > eps. \tag{4.7}$$

В противном случае система (4.I) – (4.3) с заданной точностью считается плохо обусловленной системой. Используя соотношения (4.4) последовательно при $i = M-n+I, \ldots, 1, 0$, выделяем $y_{M-n+I}, \ldots, y_1, y_0$. Относительно достаточных условий устойчивости рассматриваемого метода см. [24].

2. Программная реализация

Имя процедуры и совокупность формальных параметров имеет вид:

clonf (m, n, ko, uo, um, y, p, f, eps, lm, lp) .

Здесь:

$m+1$ – порядок системы;

n – общее количество уравнений в (4.2) и (4.3);

ko – количество уравнений в (4.2);

массив $um[1:n-ko,\ 1:n+1]$ – расширенная матрица системы (4.3);

массив $uo[1:ko,\ 1:n]$ – расширенная матрица системы (4.2);

массив $y[0:m]$ – массив ответов;

массив $p[0:m-n,\ 1:n]$ – массив коэффициентов системы (4.1);

массив $f[0:m-n]$ – вектор правых частей системы (4.I);

eps – малая положительная величина, задаваемая пользователем;

lm – метка, на которую выходит программа в тех случаях, когда не выполняются условия (4.5);

lp – метка, на которую выходит программа в тех случаях, когда не выполняется условие (4.7).

§ 5. Алгоритм решения разностной краевой задачи методом трехточечной матричной прогонки

Алгоритм предназначен для решения систем линейных алгебраических уравнений, матрицы которых имеют трехдиагональную блочную структуру. Такие системы возникают, например, при решении одного уравнения второго порядка эллиптического типа методом конечных разностей. Данный алгоритм рассмотрен в [22]. Программа, реализующая метод, написана на языке ФОРТРАН. В пакете предусмотрены две подпрограммы, реализующие данный алгоритм: CLS2F и CLS2FM . Отличие второй подпрограммы от первой состоит в том, что она предназначена для решения систем большой размерности. Для этого в программе осуществляется последовательное генерирование коэффициентов исходной системы с помощью подпрограммы CENMAT . При этом на каждом шаге вычисления прогоночных коэффициентов соответствующая информация записывается на диск и по мере необходимости при обратной прогонке с диска вызывается. Относительно достаточных условий устойчивости метода см. [22].

1. Описание алгоритма

Задана система алгебраических уравнений

$$\begin{aligned} &y_0 = F_0 \\ &A_i y_{i-1} - C_i y_i + B_i y_{i+1} = F_i, \quad 1 \le i \le M-1, \\ &y_M = F_M . \end{aligned} \tag{5.1}$$

Здесь y_i, F_0, F_i, F_M – столбцы высоты N, A_i, C_i, B_i – квадратные матрицы размера $N \times N$.

Введем следующие соотношения

$$y_i = L_i y_{i+1} + K_i, \qquad i = 1, \dots, M-1, \tag{5.2}$$

где

$$\begin{aligned} &L_1 = C_1^{-1} B_1, \quad K_1 = C_1^{-1}(A_1 F_0 - F_1), \\ &L_i = (C_i - A_i L_{i-1})^{-1} B_i, \quad K_i = (C_i - A_i L_{i-1})^{-1}(A_i K_{i-1} - F_i). \end{aligned}$$

При этом для $i = M-1$ имеем

$$y_{M-1} = L_{M-1} y_M + K_{M-1} . \tag{5.4}$$

Используя граничное условие $y_M = F_M$, из (5.4) находим y_{M-1}. После этого из (5.2) определяем значения y_i, $i = M-2, \dots, 1$.

2. Программная реализация

Поскольку, как сказано выше, в пакете предусмотрены два варианта программ, рассмотрим их отдельно.

Обращение к подпрограмме решения (5.1) в первом случае имеет вид

CLS2F(M,N, NN, M1N, M2N, Y,A,C,B,F, R, EPS).

Здесь:

M – целое число, фигурирующее в (5.1);

N – целое число;

$NN = N \times N$ – целое число;

$M1N = (M+1) \times N$ – целое число;

$M2N = (M-1) \times N \times N$ – целое число;

Y – массив размерности $M1N$, в котором помещаются искомое решение системы (5.1), а также известные значения решения в граничных точках;

A, C, B – массивы размерности $M2N$, в которые пользователь помещает значения коэффициентов системы соответственно

$$A_i\ C_i\ B_i, \qquad i = 1, \dots, M-1 ;$$

F – массив размерности $M1N$, в который пользователь помещает значение y_0 (первые N ячеек), значения F_i (следующие $(M-1) \times N$ ячеек) и значение y_M (последние N ячеек);

R – вспомогательный массив размерности $N \times N$;

EPS – малая положительная величина, задаваемая пользователем. Программа решения (5.1) устроена таким образом, что если при вычислении C_1^{-1} или $(C_i - A_i L_{i-1})^{-1}$, $i = 1, \dots, M-2$, очередной ведущий элемент соответствующей матрицы станет меньше EPS, то программа выходит на метку METHOD IS BAD, и это означает, что с заданной точностью метод не подходит для решения (5.1). Если же при вычислении $(C_{M-1} - A_{M-1} L_{M-2})^{-1}$ очередной ведущий элемент станет меньше EPS, то программа выходит на метку "PROBLEM ISBAD". Это означает, что с заданной точностью система (5.1) считается плохо обусловленной системой.

Обращение ко второй подпрограмме имеет вид

CLS2FM(M,N, NN, MN, A, C, B, R, RR, F, Y1, Y2, Y, EPS).

Здесь

M – целое число;

N – целое число;

$NN = N \times N$ – целое число;

$MN = M \times N$ – целое число;

A, C, B, R, RR – массивы размерности MN;

$F, Y1, Y2$ – массивы размерности N;

Y - массив размерности MN, в котором помещается искомое решение системы (5.I);

EPS - малая положительная величина, задаваемая пользователем.

Программа CLS2FM выходит на метки "METHOD IS BAD" и "PROBLEM IS BAD" в тех же случаях, что и программа CLS2F.

Литература

I. Горбунов-Посадов М.М., Карпов В.Я., Корягин Д.А., Красотченко В.В. Мартынюк В.В. Пакет прикладных программ САФРА. Системное наполнение. Препринт ИПМ АН СССР, № 85. М., I977.

2. Еремин А.Ю., Марьяшкин Н.Я. Пакет программ SPARSE для решения систем линейных алгебраических уравнений с разреженными матрицами. Сообщения по программному обеспечению ЭВМ. М., ВЦ АН СССР, I978.

3. Еремин А.Ю., Марьяшкин Н.Я. Пакетная обработка систем линейных алгебраических уравнений с разреженными матрицами. Ж. вычисл. матем. и матем. физ., I978, т.I8, № 6, I56I-I570.

4. Еремин А.Ю., Марьяшкин Н.Я. Пакет программ SOLVER. Системы нелинейных функциональных и обыкновенных дифференциальных уравнений с разреженными якобиевыми матрицами. Сообщения по программному обеспечению ЭВМ. М., ВЦ АН СССР, I980.

5. Гельфанд И.М., Локуциевский О.В. Метод "прогонки" для решения разностных уравнений. В книге С.К.Годунова, В.С.Рябенького "Введение в теорию разностных схем". М., Физматгиз, I962, Дополнение 2, 283-309.

6. Березин И.С., Жидков Н.П. Методы вычислений, т.2. М., Физматгиз, I960, 387-390.

7. Бабушка И., Витасек Э., Прагер М. Численные процессы решения дифференциальных уравнений. М., "Мир", I969, I2I-I52.

8. Бахвалов Н.С. Численные методы. М., "Наука", I973.

9. Дегтярев Л.М., Фаворский А.П. Потоковый вариант метода прогонки. Ж. вычисл. матем. и матем. физ., I968, т.8, № 3, 679-684.

I0. Абрамов А.А., Андреев В.Б. О применении метода прогонки к нахождению периодических решений дифференциальных и разностных уравнений. Ж. вычисл.матем. и матем. физ., I963, т.3, № 2, 377-38I.

II. Абрамов А.А. Вариант метода прогонки. Ж.вычисл.матем. и матем.физ., I96I, т.I, № 2, 349-35I.

I2. Биргер Е.С., Конюхова Н.Б. Численный расчет распространения радиоволн в вертикально-неоднородной тропосфере. Радиотехника и электроника, I969, т.XIУ, № 7, II47-II56.

I3. Полак Э. Численные методы оптимизации. М., "Мир", I974.

I4. T.H. Barret. A Prüfer transformation for matrix differential equations. Proc. Amer. Math. Soc., 1957, vol.8, N3, 510-518.

I5. Лидский В.Б., Нейгауз М.Г. К методу прогонки в случае самосопряженной системы второго порядка. Ж.вычисл.матем. и матем.физ., I962, т.2, № I, I6I-I65.

I6. Тауфер И. Решение граничных задач для систем линейных дифференциальных уравнений. М., "Наука", I98I.

I7. Валишвили Н.В. Методы расчета оболочек вращения на ЭВЦМ.М., "Машиностроение", I976.

I8. Абрамов А.А. О переносе граничных условий для систем линейных обыкновенных дифференциальных уравнений (вариант метода прогонки). Ж.вычисл.матем. и матем.физ., I96I, т.I, № 3, 542-545.

I9. Ульянова В.И. О переносе неоднородных граничных условий и о вычислении собственных функций. Ж.вычисл.матем. и матем.физ., I976, т.I6, № 4, I057-I059.

20. Годунов С.К. О численном решении краевых задач для систем линейных обыкновенных дифференциальных уравнений. Успехи матем.наук, I96I, т.I6, № 3(99), I7I-I74.

21. K. Moszyński. A method of solving the boundary value problem for a system of linear ordinary differential equations. Algorytmy, 1964, 11, N3, 25-43.

22. Огнева В.В. Метод "прогонки" для решения разностных уравнений. Ж. вычисл.матем. и матем.физ., I967, т.4, № 4, 803-8I2.

23. Дегтярев Л.М., Фаворский А.П. Потоковый вариант метода прогонки для разностных задач с сильно меняющимися коэффициентами. Ж. вычисл.матем. и матем.физ., I969, т.9, № I, 22I-2I8.

24. Сафонов Н.Д. О методе прогонки для решения разностных краевых задач. Ж.вычисл.матем. и матем.физ., I964, т.4, № 2, 256-266.

25. Левшенко Е.В., Ульянова В.И. Алгоритмы решения линейных самосопряженных краевых задач и задач с условиями периодичности для обыкновенных дифференциальных уравнений второго порядка методами классической дифференциальной прогонки. Информ.бюлл. "Алгоритмы и программы" ВНТИЦентра, I979, № 2, П0034I3.

26. Заболоцкая А.Ф. Алгоритм решения самосопряженных двухточечных линейных краевых задач для одного дифференциального уравнения второго порядка (потоковая прогонка). Информ.бюлл. "Алгоритмы и программы" ВНТИЦентра, 1979, № 2, П003475.
27. Конюхова Н.Б., Ульянова В.И. Алгоритмы решения линейных краевых задач для обыкновенных дифференциальных уравнений второго порядка с вещественными и комплексными коэффициентами методами ортогональной дифференциальной прогонки. Информ.бюлл. "Алгоритмы и программы" ВНТИЦентра, 1979, № 2, П003409.
28. Конюхова Н.Б., Левшенко Е.В. Алгоритм решения самосопряженной двухточечной краевой задачи для систем линейных обыкновенных дифференциальных уравнений второго порядка с вещественными коэффициентами (матричная классическая дифференциальная прогонка). Информ. бюлл. "Алгоритмы и программы" ВНТИЦентра, 1979, № 2, П003442.
29. Заболоцкая А.Ф. Алгоритм решения самосопряженной двухточечной линейной краевой задачи для системы линейных обыкновенных дифференциальных уравнений второго порядка с вещественными коэффициентами. Информ.бюлл. "Алгоритмы и программы" ВНТИЦентра, 1979, № 5, П003853.
30. Дышко А.Л. Алгоритм метода прогонки в случае самосопряженной системы дифференциальных уравнений второго порядка с комплексными коэффициентами. Информ.бюлл. "Алгоритмы и програмы" ВНТИЦентра, 1979, № 2, П003474.
31. Воробьев В.Н. Алгоритм решения двухточечных краевых задач методом классической прогонки для систем линейных обыкновенных дифференциальных уравнений первого порядка (*CLS1D*). Информ.бюлл. "Алгоритмы и программы" ВНТИЦентра, 1978, № 5, П003125.
32. Абрамов А.А. Алгоритм решения краевых задач для систем линейных обыкновенных дифференциальных уравнений методом ортогонального переноса граничных условий. Информ.бюлл. "Алгоритмы и программы" ВНТИЦентра, 1979, № 2, П003412.
33. Парийский Б.С. Алгоритм решения краевых задач для систем линейных обыкновенных дифференциальных уравнений методом дифференциальной ортогональной прогонки. Информ.бюлл. "Алгоритмы и программы" ВНТИЦентра, 1979, № 2, П003492.
34. Абрамов А.А., Ульянова В.И. Алгоритм решения краевых задач для систем линейных обыкновенных дифференциальных уравнений модифицированным методом ортогонального переноса граничных условий.Информ. бюлл. "Алгоритмы и программы" ВНТИЦентра, 1979, № 2, П003476.
35. Бураго Н.Г., Любимов В.М. Алгоритм дифференциальной прогонки с промежуточной ортогонализацией и нормировкой базисных решений для

систем обыкновенных линейных дифференциальных уравнений. Информ. бюлл. "Алгоритмы и программы" ВНТИЦентра, 1979, № 4, П003713.
36. Парийский Б.С. Алгоритм решения двухточечных краевых задач для систем линейных дифференциальных уравнений с нераспадающимися краевыми условиями. Информ.бюлл. "Алгоритмы и программы"ВНТИЦентра, 1979, № 2, П003493.
37. Диткин В.В. Алгоритмы решения разностных краевых задач методом трехточечных прогонок. Информ.бюлл. "Алгоритмы и программы" ВНТИЦентра, 1979, № 3, П003554.
38. Диткин В.В. Алгоритм решения разностной краевой задачи методом трехточечной матричной прогонки. Информ.бюлл. "Алгоритмы и программы" ВНТИЦентра, 1980, № 6, П004609.
39. Дышко А.Л. Потоковый вариант метода прогонки для разностных задач с сильно меняющимися коэффициентами. Информ.бюлл. "Алгоритмы и программы" ВНТИЦентра, 1979, № 3, П003638.
40. Чечель И.И. Алгоритм метода прогонки к нахождению периодических решений разностных уравнений. Информ.бюлл. "Алгоритмы и программы" ВНТИЦентра, 1979, № 2, П003490.
41. Чечель И.И. Алгоритм метода прогонки для решения разностных краевых задач. Информ.бюлл. "Алгоритмы и программы" ВНТИЦентра, 1979, № 2, П003489.
42. Монастырный П.И., Басик В.А. О численном решении краевых задач, для которых не выполняется принцип максимума. Изв.АН БССР, серия физ.-мат.наук, 1971, № 2, 28-35.
43. Монастырный П.И., Басик В.А. О методе прогонки в случае многомерных разностных уравнений. Изв.АН БССР, серия физ.-мат.наук, 1968, № 4, 63-71.
44. Монастырный П.И. О методе прогонки для систем второго порядка. Ж. вычисл.матем. и матем.физ., 1971, т.11, № 4, 925-933.
45. Монастырный П.И. Об одном аналоге метода А.А.Абрамова. Ж.вычисл. матем.и матем.физ., 1965, т.15, № 2, 342-345.
46. Монастырный П.И. Граничные задачи с неразделенными условиями для систем второго порядка и метод прогонки. Изв.АН БССР, серия физ.-мат.наук, 1977, № 5, 32-38.
47. Дж.Бэкус и др. Алгоритмический язык АЛГОЛ-60. Пересмотренное сообщение, пер.с англ. М., "Мир", 1965, 62.
48. A.A. Abramov, E.S. Birger, N.B. Konyukhova, V.I Ulyanova. On methods of numerical solution of boundary value problems for systems of

linear ordinary differential equations. Colloquia Math. Soc. Janos Bolyai, 22, Numer. Methods. Keszthely (Hungary). Amsterdam - - Oxford - New York, North - Holland Publ. Co., 1979, 33-67.

49. Абрамов А.А., Диткин В.В., Конюхова Н.Б., Партийский Б.С., Ульянова В.И. Вычисление собственных значений и собственных функций обыкновенных дифференциальных уравнений с особенностями. Ж.вычисл. матем. и матем.физ., 1980, т.20, № 5, 1155-1173.
50. Биргер Е.С. Алгоритм переноса граничных условий для систем линейных обыкновенных дифференциальных уравнений. В сб. "Алгоритмы и алгоритмические языки", М., ВЦ АН СССР, 1973, вып.6, 3-14.
51. Caffrey J., Alg. 66 INVRS, "Comm. ACM", 4, 7, 322 (1961).
52. Парийский Б.С. Некоторые приемы экономии памяти ЭВМ или времени счета для различных вариантов метода дифференциальной прогонки. Ж.вычисл.матем. и матем.физ., 1980, т.20, № 1, 69-76.
53. Любимов Ю.К., Газиева А.А. Алгоритмы метода Монте-Карло и численного интегрирования дифференциальных уравнений с автоматическим выбором шага. М., ОТИИ, 1970.
54. Фаддеев Д.К., Фаддеева В.Н. Вычислительные методы линейной алгебры. М., Физматгиз, 1960.

Содержание

Пакет прикладных программ для решения линейных
двухточечных краевых задач

Т-04438. Подписано в печать 9/III-82г. Зак.65. Тир. 301 экз.
Уч.-изд. л.2,18. Усл. печ. л. 4. Формат бумаги 60x90 1/16.
Цена 20 коп.

Отпечатано на ротапринтах в ВЦ АН СССР
Москва, В-333, ул. Вавилова, 40