ЖУРНАЛ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И МАТЕМАТИЧЕСКОЙ ФИЗИКИ, 2020, том 60, № 7, с. 1111–1125

УДК 519.622

ПОСТРОЕНИЕ И АНАЛИЗ ЯВНЫХ АДАПТИВНЫХ ОДНОШАГОВЫХ МЕТОДОВ ЧИСЛЕННОГО РЕШЕНИЯ ЖЕСТКИХ ЗАДАЧ

© 2020 г. Л. М. Скворцов

105005 Москва, 2-я Бауманская, 5, МГТУ им. Н.Э. Баумана, Россия

e-mail: lm_skvo@rambler.ru

Поступила в редакцию 29.12.2018 г.
Переработанный вариант 26.12.2019 г.
Принята к публикации 10.03.2020 г.

Рассматривается построение адаптивных методов, основанных на явных стадиях Рунге–Кутты. Коэффициенты этих методов настраиваются на решаемую задачу, с помощью покомпонентных оценок наибольших по модулю собственных значений матрицы Якоби. Такие оценки нетрудно получить по результатам стадий явного метода, что практически не требует дополнительных вычислений. В работе исследуется влияние вычислительных ошибок и жесткости решаемой задачи на устойчивость и точность численного решения. Проведенный анализ позволяет построить эффективные явные методы, не уступающие неявным методам при решении многих жестких задач. Предложены новые вложенные пары адаптивных методов и приведены результаты численных экспериментов. Библ. 24. Фиг. 4. Табл. 6.

Ключевые слова: обыкновенные дифференциальные уравнения, жесткая задача Коши, явные адаптивные методы.

DOI: 10.31857/S0044466920070108

1. ВВЕДЕНИЕ

Ограничение на размер шага не позволяет использовать традиционные явные методы Рунге–Кутты и Адамса для эффективного решения жестких задач. Поэтому для решения таких задач обычно применяют неявные методы. В то же время неявные методы требуют вычисления матрицы Якоби и решения системы нелинейных алгебраических уравнений, что усложняет их реализацию и затрудняет использование при наличии сложных нелинейных зависимостей. Поэтому наряду с неявными методами разрабатывают и применяют специальные явные методы, позволяющие эффективно решать жесткие задачи.

Класс жестких задач настолько разнообразен, что существует множество определений таких задач, но среди них нет общепринятого. Авторы известной книги по численному решению жестких задач [1] Э. Хайрер и Г. Ваннер считают, что наиболее практичным определением понятия “жесткий” является самое раннее, данное в 1952 г. Кертиссем и Хиршфельдером [2]: “Жесткие уравнения – это уравнения, для которых определенные неявные методы, в частности ФДН, дают лучший результат, обычно несравненно более хороший, чем явные методы”. В настоящее время известны специальные явные методы, эффективные для многих жестких задач, поэтому под явными методами в этом определении следует понимать классические явные методы Рунге–Кутты и Адамса. Отметим, что термины “жесткие уравнения” или “жесткая задача” применимы к конкретной задаче Коши, т.е. при заданных начальных условиях и интервале интегрирования, поскольку задача, жесткая на большом интервале, может быть нежесткой на меньшем интервале или при других начальных условиях.

Будем решать задачу Коши для системы ОДУ

$$\mathbf{y}' = \mathbf{f}(t, \mathbf{y}), \quad \mathbf{y}(t_0) = \mathbf{y}_0, \quad 0 \le t \le T, \tag{1.1}$$

где t – независимая переменная, $\mathbf{y}$ – вектор переменных состояния, $\mathbf{f}(t, \mathbf{y})$ – нелинейная векторная функция. Предположим, что задача (1.1) имеет единственное решение $\mathbf{y}(t)$, а функция $\mathbf{f}$ –

гладкая в любой точке решения на интервале интегрирования. Тогда на всем интервале определена и непрерывна матрица Якоби системы (1.1)

$$\mathbf{J}(t) = \frac{\partial \mathbf{f}(t, \mathbf{y}(t))}{\partial \mathbf{y}}.$$

Требование гладкости правой части не всегда согласуется с реальными задачами, правые части которых могут содержать разрывные характеристики типа реле или переключателей. При наличии таких элементов будем предполагать, что число переключений конечно, а весь интервал интегрирования можно разбить на несколько интервалов, на каждом из которых функция **f** остается гладкой. Тогда решения "склеиваются", т.е. на каждом последующем интервале в качестве начального условия принимается решение, полученное в конце предыдущего интервала.

Для дальнейшего изложения нам нужно определить меру жесткости задачи Коши (1.1). В качестве такой меры можно использовать минимальное число шагов, необходимое для устойчивого решения задачи простейшим явным методом. Рассмотрим сначала линейную задачу

$$\mathbf{y}' = \mathbf{J}\mathbf{y}, \quad \mathbf{y}(t_0) = \mathbf{y}_0, \quad 0 \le t \le T, \tag{1.2}$$

где матрица **J** имеет собственные значения $\lambda_1, \ldots, \lambda_n$ с отрицательными действительными частями. Для такой задачи число шагов пропорционально интервалу интегрирования T и обратно пропорционально минимальной постоянной времени $\tau_{\min}$. Поэтому для линейных задач вычислительные затраты можно оценить числом

$$M_{\text{stf}} = \mu T, \quad \mu = 1/\tau_{\min} = \max_i \operatorname{Re}(-\lambda_i).$$

Значение M_{stf}, округленное до большего целого, – минимальное число шагов явного метода Эйлера, обеспечивающее устойчивое решение всех задач вида (1.2), удовлетворяющих условиям $\operatorname{Re}(\lambda_i) < 0$, $|\operatorname{Im}(\lambda_i)| < |\operatorname{Re}(\lambda_i)|$.

В случае нелинейной задачи (1.1) спектр матрицы Якоби зависит от t, поэтому вычислительные затраты явных методов можно оценить с помощью интеграла

$$M_{\text{stf}} = \int_0^T \max\left(\max_i \operatorname{Re}(-\lambda_i(t)), 0\right) dt. \tag{1.3}$$

Величину M_{stf} назовем *мерой жесткости* задачи Коши. В формуле (1.3) оцениваются вычислительные затраты, вызванные только жесткостью задачи. Реальное число шагов может превышать эти оценки вследствие негладкости правой части, наличия больших собственных значений вблизи мнимой оси или в правой полуплоскости (колебательные и плохо обусловленные задачи), а также по другим причинам.

Аналогично можно определить меры колебательности и неустойчивости задачи Коши. Величину

$$M_{\text{osc}} = \int_0^T \max_i \operatorname{Im}(\lambda_i(t))\, dt$$

назовем *мерой колебательности*, а величину

$$M_{\text{inst}} = \int_0^T \max\left(\max_i \operatorname{Re}(\lambda_i(t)), 0\right) dt$$

назовем *мерой неустойчивости* задачи Коши. Значения этих мер, а также число переменных n и величина интервала интегрирования T для восьми жестких тестов из [1] приведены в табл. 1. Отметим, что задача BEAM имеет чисто мнимый спектр матрицы Якоби. Поэтому формально ее можно отнести к колебательным задачам, но она проявляет свойство жесткости, поскольку эффективно может быть решена только неявными методами. Этот факт еще раз подтверждает трудность формального определения класса жестких задач.

Наряду с неявными методами, для решения жестких задач с вещественным жестким спектром успешно применяют специальные явные методы. Принципы построения таких методов рассмотрим на примере явного двухстадийного метода Рунге–Кутты 1-го порядка с функцией устойчивости $R(z) = 1 + z + dz^2$. На фиг. 1 приведены области устойчивости такого метода при

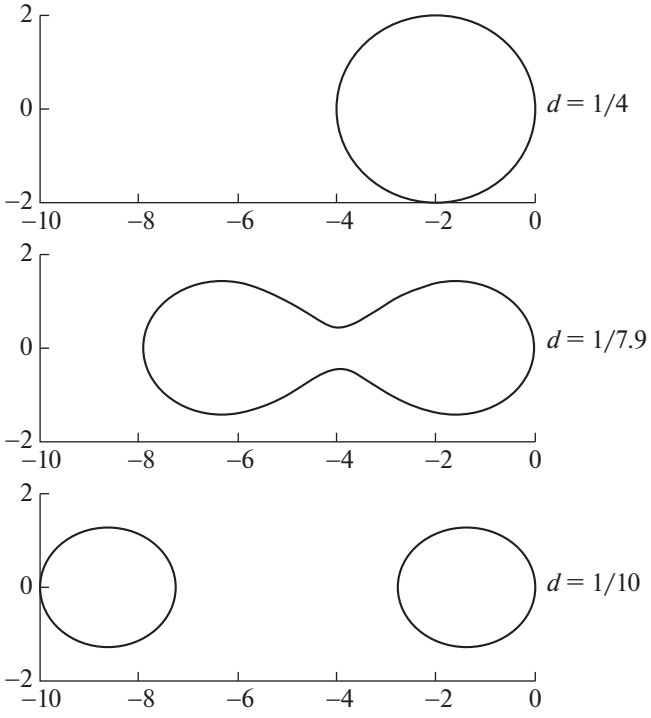
Таблица 1. Характеристики тестовых задач

Задача	n	T	M_{stf}	M_{osc}	M_{inst}
VDPOL	2	2	3.84×10^6	4.0	35.8
ROBER	3	10^4	8.07×10^7	0	0
OREGO	3	360	1.13×10^7	1.5	27.1
HIRES	8	321.8122	3.44×10^4	0.006	0
PLATE	80	7	6.96×10^3	1.02×10^4	0
BEAM	80	5	0	3.2×10^4	0
CUSP	96	1.1	6.87×10^4	1.8	21.0
BRUSS	1000	10	2×10^5	0	0

различных значениях d. При $d > 1/8$ область устойчивости односвязная, а ее длина равна $1/d$. Уменьшая d, можно увеличить длину области устойчивости. При $d < 1/8$ область устойчивости перестает быть односвязной и состоит из двух разделенных областей, наиболее удаленная из которых позволяет обеспечить стабилизацию метода в жесткой части спектра.

Приведенные области позволяют сформулировать два способа построения явных методов для жестких задач. Первый способ основан на максимальном расширении области устойчивости. Построение многочленов устойчивости одношаговых методов такого типа обычно выполняют исходя из условия чебышёвского альтернанса, поэтому их часто называют методами Рунге–Кутты–Чебышёва. Методы с расширенными областями устойчивости рассматривались в [1], [3]–[7] и многих других работах.

Идея второго способа заключается в получении на основе предварительных стадий оценок наибольших по модулю собственных значений матрицы Якоби, которые используются в заключительной формуле для стабилизации расчетной схемы в полученных точках жесткого спектра. Как отмечалось в [7]–[9], эта идея (обычно в замаскированном виде) использовалась во многих явных нелинейных методах (например, в методах из [10]–[14]). В [10] впервые были четко выде-



Фиг. 1.

лены этапы получения оценок собственных значений и последующего использования этих оценок для стабилизации численного решения. Несмотря на большое число работ, посвященных явным нелинейным методам, они так и не получили широкого практического применения. В [14] на конкретных примерах было показано, что предложенные ранее нелинейные методы пригодны для решения с низкой точностью лишь весьма узкого класса жестких задач. Основной причиной низкой эффективности этих методов является снижение реального порядка при решении жестких задач. Например, двухстадийный метод имеет 2-й порядок при решении нежестких задач, но при решении жестких задач реальный порядок снижается до нулевого (см. [15], [16]). Под реальным порядком мы понимаем порядок, который оценивается при реальных размерах шага h, позволяющих обеспечить заданную точность, тогда как классический порядок оценивается при $h \to 0$.

Более эффективные явные нелинейные одношаговые и многошаговые методы, названные адаптивными, были предложены в [15], [16]. Рассмотренные в этих работах одношаговые методы позволяют обеспечить 1-й порядок для жестких задач при числе стадий $s = 3$ и 2-й порядок при $s > 3$. Общий принцип построения адаптивных одношаговых методов изложен в [7]–[9], а в [8] предложен метод, имеющий для жестких задач 3-й порядок. Столь же эффективные для жестких задач одношаговые методы более высоких порядков построить не удалось, но аналогичные многошаговые методы могут иметь практически любой порядок, и на их основе в [17] построен явный адаптивный метод переменного порядка. Явные адаптивные методы реализованы в программных комплексах МВТУ [18], [19] и SimInTech [20], а также в виде программ в системе MathCad, которые доступны в дополнительных материалах к книге [7] по адресу: http://3v-services.com/books/978-5-97060-636-0. В настоящей статье исследуются явные адаптивные одношаговые методы, которые заметно проще многошаговых методов и в то же время позволяют эффективно решать многие жесткие задачи при умеренных требованиях к точности.

Оценим вычислительные затраты явных методов в зависимости от жесткости при решении задач с вещественным жестким спектром. Определим вычислительные затраты как число вычислений правой части на всем интервале Nf. Тогда $Nf = N \times s$, где N – число выполненных шагов, s – число стадий на одном шаге. Для методов с односвязной областью устойчивости (к ним относятся классические методы и методы с расширенными областями), исходя из условия устойчивости численного решения, получаем

$$Nf > cM_{\text{stf}}, \quad c = s/l(s),$$

где $l(s)$ – длина области устойчивости вдоль вещественной оси. При числе стадий, равном порядку метода, для метода 2-го порядка имеем $c = 1$, для метода 3-го порядка $c = 1.19$ и для метода 4-го порядка $c = 1.44$. Таким образом, вычислительные затраты классических методов пропорциональны жесткости задачи M_{stf}.

Оценим теперь затраты чебышёвских методов. Для них размер шага h не ограничен условием устойчивости, а выбирается исходя из заданной точности, и может быть таким же большим, как в неявных методах. Но при этом увеличивается число стадий s, которое может достигать нескольких сотен и определяется из условия устойчивости $l(s) > h \max\limits_i (-\lambda_i)$, где $l(s) \approx Ks^2$ (для методов 2-го порядка $K \approx 0.8$, а для методов 3-го порядка $K \approx 0.48$). Тогда оценку затрат получаем в виде

$$Nf > c\sqrt{M_{\text{stf}}}, \quad c = \sqrt{N/K}.$$

Если задача не очень жесткая, то число шагов N зависит от характера решения и почти не зависит от жесткости, поэтому вычислительные затраты пропорциональны $\sqrt{M_{\text{stf}}}$.

Методы с расширенными областями устойчивости применяют для решения задач с распределенным вещественным спектром матрицы Якоби, к которым относятся, прежде всего, задачи, полученные в результате дискретизации уравнений в частных производных параболического типа (например, задача BRUSS из [1]). В таких задачах жесткие составляющие, соответствующие различным собственным значениям, сильно связаны между собой. Другой тип задач с вещественным спектром – когда жесткие составляющие слабо связаны между собой. Это проявляется в том, что для получения с необходимой точностью оценок всех жестких собственных значений матрицы Якоби достаточно выполнить несколько итераций покомпонентного степенного метода. К таким задачам относятся задачи VDPOL, ROBER, OREGO, HIRES и CUSP из табл. 1. Именно при решении таких задач явные адаптивные методы наиболее эффективны и при расче-

тах с двойной точностью позволяют эффективно решать задачи с жесткостью до $M_{\text{stf}} = 10^{155}$ (см. результаты решения задачи Капса в [7], [9]). Никакие другие явные методы не могут решать такие задачи.

2. ПОСТРОЕНИЕ ЯВНЫХ АДАПТИВНЫХ МЕТОДОВ

В общем случае явные адаптивные одношаговые методы строятся на основе явных стадий Рунге–Кутты, выполняемых по формулам

$$\mathbf{Y}_i = \mathbf{y}_0 + h\sum_{j=1}^{i-1} a_{ij}\mathbf{F}_j, \quad \mathbf{F}_i = \mathbf{f}\left(t_0 + c_i h, \mathbf{Y}_i\right), \quad i = 1,\ldots,s. \tag{2.1}$$

Пусть $a_{i,i-1} \neq 0$, $i = 2,\ldots,s$. Определим векторы $\mathbf{u}_1,\ldots,\mathbf{u}_s$ в виде

$$\mathbf{u}_i = \frac{1}{\beta_{ii}}\left(\mathbf{F}_i - \sum_{j=1}^{i-1}\beta_{ij}\mathbf{u}_j\right), \quad i = 1, \ldots, s,$$

где $[\beta_{ij}] = [\mathbf{A}^0\mathbf{e}, \mathbf{A}^1\mathbf{e},\ldots,\mathbf{A}^{s-1}\mathbf{e}]$, $\mathbf{A} = [a_{ij}]$, $\mathbf{e} = [1,\ldots,1]^{\mathrm{T}}$. Тогда при решении линейной системы (1.2) будут выполняться соотношения

$$\mathbf{u}_i = h^{-1}(h\mathbf{J})^i\mathbf{y}_0, \quad i = 1, \ldots, s,$$

которые позволяют воспользоваться степенным методом для получения вектора покомпонентных оценок больших по модулю собственных значений матрицы $h\mathbf{J}$ в виде

$$\tilde{\mathbf{z}} = \mathbf{u}_s/\mathbf{u}_{s-1}. \tag{2.2}$$

В этой и последующих формулах предполагаем покомпонентное выполнение операций с векторами.

Шаг интегрирования выполняем согласно формуле

$$\mathbf{y}_1 = \mathbf{y}_0 + h\left(\sum_{i=1}^{s-1}\mathbf{u}_i/i! + \mathbf{d}_s\mathbf{u}_s\right), \tag{2.3}$$

где $\mathbf{d}_s$ – вектор настраиваемых параметров. Учитывая (2.2), эту же формулу можно записать в виде

$$\mathbf{y}_1 = \mathbf{y}_0 + h\left(\sum_{i=1}^{s-2}\mathbf{u}_i/i! + \mathbf{d}_{s-1}\mathbf{u}_{s-1}\right), \tag{2.4}$$

где $\mathbf{d}_{s-1} = \mathbf{e}/(s-1)! + \mathbf{d}_s\tilde{\mathbf{z}}$. В дальнейшем будем использовать формулу (2.4), которая содержит меньше членов и менее чувствительна к вычислительным ошибкам. Вектор $\mathbf{d}_{s-1}$ задаем из условия, чтобы численное решение уравнения $y' = \lambda y$ выражалось формулой $y_1 = Q(z)y_0$, где $z = h\lambda$, $Q(z)$ – *скалярная функция устойчивости*. Тогда для вычисления очередной компоненты вектора $\mathbf{d}_{s-1}$ (обозначим ее d_{s-1}) можно использовать формулы

$$d_0 = Q(\tilde{z}), \quad d_{i+1} = (d_i - 1/i!)/\tilde{z}, \quad i = 0, 1, \ldots, s-2. \tag{2.5}$$

В случае линейной системы (1.2) получаем

$$\mathbf{y}_1 = \mathbf{R}(\mathbf{Z})\mathbf{y}_0, \quad \mathbf{Z} = h\mathbf{J},$$

где

$$\mathbf{R}(\mathbf{Z}) = \mathbf{I} + \mathbf{Z} + \cdots + \mathbf{Z}^{s-2}/(s-2)! + \mathbf{D}_{s-1}\mathbf{Z}^{s-1}$$

есть *матричная функция устойчивости*, $\mathbf{D}_{s-1} = \operatorname{diag}(\mathbf{d}_{s-1})$.

Скалярную функцию устойчивости $Q(z)$ задаем из условий заданного порядка сходимости нежестких компонент, затухания жестких компонент, расхождения неустойчивых компонент и ограниченности коэффициента d_{s-1} (см. [7]–[9]). Анализ численных экспериментов показал, что для больших по модулю значений z порядок согласованности с экспонентой неважен. Главное –

обеспечить качественно правильное поведение (быстрое затухание либо расхождение) соответствующих компонент. Например, при $s = 3$ можно задать

$$Q(z) = \begin{cases} 1 + z + \dfrac{z^2}{2} + \dfrac{z^3}{6}, & |z| \le 1.6, \\ 0, & z < -1.6, \\ 1 + \dfrac{167}{75} z, & z > 1.6, \end{cases} \tag{2.6}$$

а при $s = 4$ можно задать

$$Q(z) = \begin{cases} 1 + z + \dfrac{z^2}{2} + \dfrac{z^3}{6} + \dfrac{z^4}{48}, & |z| \le 4.5, \\ 0, & z < -4.5, \\ 1 + z + \dfrac{107}{64} z^2, & z > 4.5. \end{cases} \tag{2.7}$$

Вычисление коэффициентов вектора $\mathbf{d}_{s-1}$ организуем таким образом, чтобы избежать операций с большими числами и деления на 0. Рассмотрим, например, вычисление d_2 при $s = 3$ и $Q(z)$ в виде (2.6). Если очередная компонента нежесткая, т.е. если $|u_3| \le 1.6|u_2|$, то вычисляем $\tilde{z} = u_3/u_2$, $d_2 = 1/2 + \tilde{z}/6$ (при $u_2 = 0$ принимаем $\tilde{z} = 0$), а иначе вычисляем

$$\tilde{z}^{-1} = u_2/u_3, \quad d_2 = \begin{cases} -\tilde{z}^{-1} - \tilde{z}^{-2}, & \tilde{z}^{-1} < 0, \\ \dfrac{92}{75} \tilde{z}^{-1}, & \tilde{z}^{-1} > 0. \end{cases}$$

Согласно (2.1), первая стадия следующего шага выполняется по формуле

$$\mathbf{F}_1 = \mathbf{f}_1 = \mathbf{f}(t_0 + h, \mathbf{y}_1). \tag{2.8}$$

В [8], [9] предложена модификация адаптивных методов, при которой первая стадия на всех шагах, кроме первого, выполняется согласно формуле

$$\mathbf{F}_1 = \mathbf{f}_1 = \mathbf{f}_0 + \sum_{i=1}^{s-2} \mathbf{u}_{i+1}/i! + \mathbf{d}_{s-1}\mathbf{u}_s. \tag{2.9}$$

Из (2.2), (2.5) следует, что формулу (2.9) можно записать также в виде

$$\mathbf{F}_1 = \mathbf{f}_1 = \mathbf{f}_0 + \sum_{i=1}^{s-3} \mathbf{u}_{i+1}/i! + \mathbf{d}_{s-2}\mathbf{u}_{s-1}. \tag{2.10}$$

Использование формулы (2.10) вместо (2.8) позволяет сэкономить на каждом шаге одно вычисление правой части, а также уменьшает влияние на устойчивость неточности оценок собственных чисел. Поэтому стадия в виде (2.10) была названа *стабилизированной первой стадией*.

Другой способ повышения устойчивости заключается в выполнении дополнительной коррекции жестких компонент. Например, при $s = 3$ коррекцию жесткой компоненты можно выполнять по формуле

$$y_{1\,\mathrm{cor}} = y_0 + hd_1 f_0 + (1 - d_1)(y_1 - y_0) + hd_2 (f_1 - f_0) \tag{2.11}$$

(индекс компоненты опускаем). В результате появляется дополнительный корень многочлена устойчивости, близкий к оценке $\tilde{z}$ для этой компоненты, что приводит к расширению области устойчивости в окрестности $\tilde{z}$. Отметим, что коррекцию (2.11) можно применять итерационно, а если применить ее к методу Эйлера $\mathbf{y}_1 = \mathbf{y}_0 + h\mathbf{f}_0$, то получим формулу адаптивного метода (2.4).

Таким образом, можно выделить три типа адаптивных методов: обычный, со стабилизированной 1-й стадией, с коррекцией. Рассмотрим особенности каждого из этих типов, их преимущества и недостатки.

3. УСТОЙЧИВОСТЬ АДАПТИВНЫХ МЕТОДОВ

В [15], [16] были предложены явные адаптивные методы, стадии которых выполняются согласно формулам

$$\begin{gathered}\mathbf{F}_1 = \mathbf{f}(t_0, \mathbf{y}_0), \quad \mathbf{Y}_2 = \mathbf{y}_0 + h\beta\mathbf{F}_1, \quad \mathbf{F}_2 = \mathbf{f}(t_0 + h\beta, \mathbf{Y}_2),\\ \mathbf{Y}_i = \mathbf{y}_0 + h[(\beta - \alpha)\mathbf{F}_1 + \alpha\mathbf{F}_{i-1}], \quad \mathbf{F}_i = \mathbf{f}(t_0 + h\beta, \mathbf{Y}_i), \quad i = 3, \ldots, s.\end{gathered} \tag{3.1}$$

Исходя из классической теории, оптимальное значение β в этих методах равно $2/3$. В этом случае при $s \geq 3$ метод может иметь 3-й порядок. Однако анализ решения простейших модельных уравнений показал, что для жестких задач реальный порядок методов, основанных на стадиях вида (3.1), не превышает $s - 2$ и не может быть выше 2-го, а оптимальное значение β равно 1 (см. [7], [8]). Функция устойчивости i-й стадии выражается формулой

$$R_i(z) = 1 + \sum_{j=1}^{i-1} \beta\alpha^{j-1} z^j.$$

Чтобы ограничить рост внутренних функций устойчивости, следует задать достаточно малое значение α. Мы рекомендуем задавать этот параметр в виде

$$\alpha = \min\left(\bar{\alpha}, \min_i \left|\tilde{z}_i^{-1}\right| h_{\text{old}}/h\right), \tag{3.2}$$

где $\bar{\alpha} = 1/2$ или $\bar{\alpha} = 1/3$, h_{old} – размер предыдущего шага, $\tilde{z}_i$ – покомпонентные оценки вида (2.2), полученные на предыдущем шаге.

Исследуем устойчивость адаптивных методов, основанных на стадиях (3.1) при $s = 3$, в зависимости от неточности вычисления оценок $\tilde{z}_i$. Решаем уравнение Далквиста $y' = \lambda y$. Пусть $\overline{z} = h\lambda$ и оценка этой величины получена с относительной ошибкой ε, тогда $\tilde{z} = \overline{z}(1 + \varepsilon)$. Предположим также, что

$$|\varepsilon| \ll 1 \ll -\overline{z}. \tag{3.3}$$

Для обычного метода, исходя из условия $Q(\tilde{z}) = 0$, получаем $d_2 = -\tilde{z}^{-1} - \tilde{z}^{-2}$. Функция устойчивости такого метода при $z = \overline{z}$ имеет вид

$$R(\overline{z}) = 1 + \overline{z} + d_2\overline{z}^2 = \varepsilon\frac{2 + \varepsilon + \overline{z}(1 + \varepsilon)}{(1 + \varepsilon)^2},$$

и при выполнении (3.3) получаем условие устойчивости в виде ограничения на значения $\overline{z}$ в виде $|\overline{z}| < |\varepsilon|^{-1}$. Отметим, что если вместо (2.4) использовать формулу (2.3), то интервал допустимых при данной ошибке значений $\overline{z}$ заметно сузится: $|\overline{z}| < \sqrt{2}|\varepsilon|^{-1/2}$.

Рассмотрим теперь метод с дополнительной коррекцией. В этом случае функция устойчивости скорректированного метода будет имеет вид

$$R_{\text{cor}}(z) = 1 + d_1 z + (1 - d_1 + d_2 z)(R(z) - 1), \quad d_1 = -\tilde{z}^{-1},$$

откуда, учитывая (3.3), получаем $R_{\text{cor}}(\overline{z}) \approx \varepsilon^2\overline{z}$ и условие устойчивости $|\overline{z}| < |\varepsilon|^{-2}$.

Для метода со стабилизированной первой стадией формулы одного шага решения уравнения Далквиста можно записать в виде

$$\begin{gathered}y_1 = y_0 + hf_0 + d_2\left(\overline{z}(y_0 + hf_0) - hf_0\right),\\ hf_1 = hf_0 + d_1\left(\overline{z}(y_0 + hf_0) - hf_0\right), \quad \overline{z} = h\lambda.\end{gathered}$$

Эти формулы задают разностную схему с характеристическим многочленом

$$x^2 - ax + b, \quad a = \frac{2\varepsilon}{1 + \varepsilon} + \frac{\varepsilon}{\overline{z}(1 + \varepsilon)^2}, \quad b = \frac{\varepsilon}{1 + \varepsilon} + \frac{\varepsilon}{\overline{z}(1 + \varepsilon)^2}.$$

При выполнении (3.3) оба корня этого многочлена по модулю меньше 1, что обеспечивает устойчивость разностной схемы. Таким образом, неточность оценки собственного значения практически не накладывает ограничения на значение $\overline{z}$. Отметим, что если вместо (2.10) использовать

Таблица 2. Значения $|\overline{z}|_{\max}$ в зависимости от ε

s	$\tilde{p}$	ARK		ARKc		ARKs	
		nf	$|\overline{z}|_{\max}$	nf	$|\overline{z}|_{\max}$	nf	$|\overline{z}|_{\max}$
3	1	3	$|\varepsilon|^{-1}$	4	$|\varepsilon|^{-2}$	2	∞
4	2	4	$\sqrt{2}|\varepsilon|^{-1/2}$	5	$\sqrt{2}|\varepsilon|^{-1}$	3	$0.5|\varepsilon|^{-1}$
6	3	6	$6^{1/3}|\varepsilon|^{-1/3}$	7	$6^{1/3}|\varepsilon|^{-2/3}$	5	$\sqrt{6/7}|\varepsilon|^{-1/2}$

эквивалентную при точном вычислении формулу (2.9), то условие устойчивости будет таким же, как у обычного метода.

Аналогичным образом получены ограничения вида $|\overline{z}| < |\overline{z}|_{\max}$ для методов, основанных на четырех стадиях вида (3.1), и для методов, основанных на шести стадиях, приведенных в [7], [8]. Значения $|\overline{z}|_{\max}$ приведены в табл. 2, где s – число стадий, на основе которых построен адаптивный метод, $\tilde{p}$ – реальный порядок метода при решении жестких задач, nf – число вычислений правой части на одном шаге. Обозначения методов приняты в виде ARK (Adaptive Runge–Kutta) – обычный метод, ARKc – с дополнительной коррекцией, ARKs – со стабилизированной 1-й стадией.

Приведем пример, подтверждающий полученные оценки. Будем решать задачу

$$\begin{gathered} y_1' = -\mu e_1(t) + \cos t, \quad e_1(t) = y_1 - \sin t, \\ y_2' = -\mu e_2(t) - \sin t, \quad e_2(t) = y_2 - \cos t, \\ y_1(0) = 0, \quad y_2(0) = 1, \quad 0 \le t \le 2\pi, \end{gathered} \tag{3.4}$$

с точным решением $y_1(t) = \sin t$, $y_2(t) = \cos t$. Используем методы со стадиями (3.1) при $s = 3$, $\beta = 1$. При точном выполнении всех вычислений получим точные оценки собственного значения. Внесем ошибки в эти оценки в виде

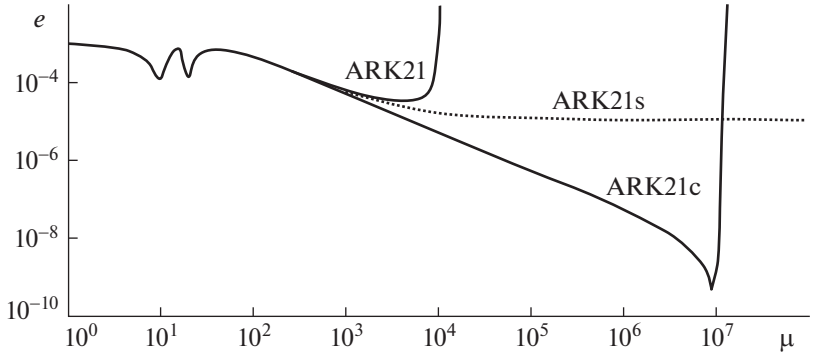
$$\tilde{z}_1 = \overline{z}_1(1 - \varepsilon), \quad \tilde{z}_2 = \overline{z}_2(1 + \varepsilon).$$

Ошибку решения вычисляем по формуле

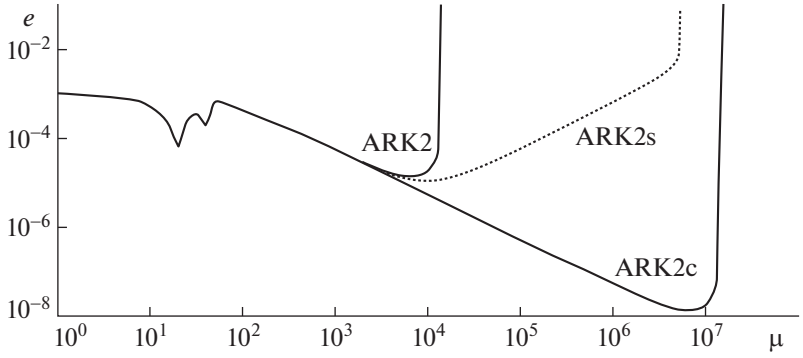
$$e = \max_t \sqrt{e_1(t)^2 + e_2(t)^2}.$$

Зависимости ошибки от жесткости задачи μ для методов рассмотренных трех типов при $h = \pi/30$ и $\varepsilon = 10^{-3}$ приведены на фиг. 2. Здесь и далее в обозначении методов первая цифра – классический порядок, а вторая – реальный порядок для жестких задач. Если эти порядки совпадают, то оставляем только одну цифру.

Как и следовало ожидать из проведенного выше анализа, расхождение решения методом ARK21 наблюдается при $h\mu > \varepsilon^{-1}$, а методом ARK21c – при $h\mu > \varepsilon^{-2}$. Внутри интервалов устойчи-



Фиг. 2.



Фиг. 3.

вости ошибки этих двух методов уменьшаются при увеличении μ, что объясняется видом уравнений и свойством жесткой точности при $\beta = 1$ (см. анализ ошибки решения уравнения Протеро–Робинсона в [7], [8]). Метод ARK21s, в отличие от ARK21, не расходится при $h\mu > \varepsilon^{-1}$, но ошибка перестает уменьшаться с ростом μ и сохраняет свое значение вплоть до $\mu = 10^{155}$, и только при $\mu = 1.4 \times 10^{155}$ выдается сообщение о переполнении.

Рассмотрим теперь методы со стадиями (3.1) при $s = 4$, $\beta = 1$, которые сохраняют 2-й порядок при решении жестких задач и обозначены через ARK2, ARK2c и ARK2s. Результаты решения задачи (3.4) этими методами при $\varepsilon = 10^{-6}$ приведены на фиг. 3. В отношении устойчивости эти результаты полностью согласуются с табл. 2. Ошибки методов ARK2 и ARK2c, как и аналогичных методов ARK21 и ARK21c, уменьшаются при увеличении μ, а ошибка метода ARK2s сначала уменьшается, а при $h\mu \approx \varepsilon^{-1/2}$ начинает возрастать. Таким образом, можно заключить, что неточность оценки собственного значения может привести к расхождению численного решения, а для методов со стабилизированной 1-й стадией – также и к снижению точности при сохранении устойчивости.

4. ТОЧНОСТЬ АДАПТИВНЫХ МЕТОДОВ

Для исследования точности используем три задачи, точные решения которых приводим ниже. Первая из них – задача Капса (см. [21]):

$$y_1' = -(\mu + 2)y_1 + \mu y_2^2, \quad y_2' = y_1 - y_2 - y_2^2, \qquad y_1(t) = \exp(-2t), \quad y_2(t) = \exp(-t), \quad 0 \le t \le 1. \tag{4.1}$$

Вторая задача – линейная неавтономная:

$$\begin{bmatrix} y_1' \\ y_2' \end{bmatrix} = \begin{bmatrix} a & b \\ b & a \end{bmatrix} \begin{bmatrix} y_1 - \sin t \\ y_2 - \cos t \end{bmatrix} + \begin{bmatrix} \cos t \\ -\sin t \end{bmatrix}, \tag{4.2}$$

$$a = -(\mu + 1)/2, \quad b = -(\mu - 1)/2, \quad y_1(t) = \sin t, \quad y_2(t) = \cos t, \quad 0 \le t \le 1.$$

Третья задача имеет вид

$$y_1' = y_2 - \frac{\mu}{2} y_1(y_1^2 + y_2^2 - 1), \quad y_2' = -y_1 - \frac{\mu}{2} y_2(y_1^2 + y_2^2 - 1), \qquad y_1(t) = \sin t, \quad y_2(t) = \cos t, \quad 0 \le t \le 1. \tag{4.3}$$

Жесткость задач определяется значением μ, которое (при больших μ) практически совпадает с мерой жесткости M_{stf}. Мы решали эти задачи с шагом $h = 1/30$ адаптивными методами, а также методом Розенброка 2-го порядка, который обозначим через Ros2 (этот метод реализован в решателе ode23s системы MATLAB, см. [22]). Ошибки решения в зависимости от μ приведены в табл. 3 для задачи (4.1), в табл. 4 для задачи (4.2) и в табл. 5 для задачи (4.3).

Таблица 3. Ошибки решения задачи (4.1)

Метод	$\mu = 1$	$\mu = 10^2$	$\mu = 10^4$	$\mu = 10^6$
ARK21	2.74×10^{-5}	2.80×10^{-4}	7.11×10^{-3}	8.28×10^{-3}
ARK21c	2.74×10^{-5}	3.67×10^{-4}	7.71×10^{-3}	8.29×10^{-3}
ARK21s	2.11×10^{-5}	8.25×10^{-4}	1.78×10^{-3}	1.20×10^{-3}
ARK2	3.02×10^{-5}	6.87×10^{-5}	9.21×10^{-5}	9.31×10^{-5}
ARK2c	3.02×10^{-5}	6.87×10^{-5}	9.13×10^{-5}	9.31×10^{-5}
ARK2s	3.01×10^{-5}	7.93×10^{-5}	2.22×10^{-4}	2.25×10^{-4}
Ros2	7.77×10^{-5}	1.85×10^{-4}	1.51×10^{-4}	1.48×10^{-4}

Таблица 4. Ошибки решения задачи (4.2)

Метод	$\mu = 1$	$\mu = 10^2$	$\mu = 10^4$	$\mu = 10^6$
ARK21	7.89×10^{-5}	1.16×10^{-3}	3.29×10^{-3}	3.33×10^{-3}
ARK21c	7.89×10^{-5}	6.29×10^{-4}	3.27×10^{-3}	3.33×10^{-3}
ARK21s	7.89×10^{-5}	4.16×10^{-3}	1.93×10^{-1}	2.13×10^{-1}
ARK2	7.92×10^{-5}	5.03×10^{-5}	2.40×10^{-5}	2.46×10^{-5}
ARK2c	7.92×10^{-5}	5.03×10^{-5}	2.37×10^{-5}	2.46×10^{-5}
ARK2s	7.92×10^{-5}	3.66×10^{-5}	7.29×10^{-5}	7.41×10^{-5}
Ros2	6.21×10^{-5}	8.94×10^{-5}	8.21×10^{-5}	8.17×10^{-5}

Таблица 5. Ошибки решения задачи (4.3)

Метод	$\mu = 1$	$\mu = 10^2$	$\mu = 10^4$	$\mu = 10^6$
ARK21	5.86×10^{-5}	2.20×10^{-4}	8.95×10^{-4}	1.05×10^{-3}
ARK21c	5.86×10^{-5}	1.85×10^{-4}	8.89×10^{-4}	9.05×10^{-4}
ARK21s	6.24×10^{-5}	4.02×10^{-4}	1.49×10^{-2}	1.59×10^{-2}
ARK2	5.86×10^{-5}	8.07×10^{-5}	9.52×10^{-4}	—
ARK2c	5.86×10^{-5}	8.07×10^{-5}	3.58×10^{-4}	—
ARK2s	5.86×10^{-5}	8.04×10^{-5}	3.58×10^{-4}	3.06×10^{-4}
Ros2	7.37×10^{-5}	1.84×10^{-4}	1.44×10^{-2}	3.44×10^{-1}

При решении задачи (4.1) все методы показали приемлемые и вполне объяснимые результаты. При повышении жесткости методы ARK21, ARK21c и ARK21s заметно снижают точность, что объясняется снижением реального порядка до 1-го. Снижение порядка подтверждается расчетами с уменьшением размера шага. Для всех методов при $\mu > 10^4$ дальнейшее повышение жесткости уже не приводит к заметному увеличению ошибки, которая мало изменяется вплоть до значения μ, при котором решение расходится.

Задача (4.2) оказалась трудной для метода ARK21s, который показывает неудовлетворительную точность при $\mu \geq 10^4$, хотя и сохраняет устойчивость вплоть до значения $\mu = 10^{155}$. Но при этом сходимость обеспечивается только при малых размерах шага. Например, при $\mu = 10^6$ и уменьшении шага в 10 раз ошибка мало изменилась и равна 0.195. И только при $h < 10^{-3}$ ошибка пропорциональна h^2. Приведенный пример показывает, что при решении некоторых жестких задач метод со стабилизированной 1-й стадией может быть менее точным по сравнению с другими адаптивными методами.

Задача (4.3) – пример жесткой задачи, трудной для решения неявными методами. При больших значениях μ все неявные методы, которые мы использовали для решения этой задачи, тре-

бовали больших вычислительных затрат и не обеспечивали требуемой точности. Причиной низкой эффективности неявных методов является медленная сходимость простых ньютоновских итераций, которые используются в этих методах. Явные адаптивные методы были более эффективными. Несколько неожиданно методы ARK21 и ARK21c имели 2-й порядок при больших значениях μ, хотя при решении других жестких задач реальный порядок снижается до 1-го. Метод ARK2 не позволил получить решение с данным шагом при $\mu > 10^4$, а ARK2c – при $\mu > 2\times 10^5$.

При решении всех трех задач методы ARK и ARKc показали близкие результаты, но при этом методы ARKc позволяют решать более жесткие задачи. Например, задача (4.2) была успешно решена методом ARK21 вплоть до значения $\mu = 10^{18}$, а методом ARK21c – вплоть до $\mu = 10^{30}$.

5. ПОСТРОЕНИЕ МЕТОДОВ ARK32 И ARK32C

Методы, построенные на основе стадий (3.1) при $s = 3$, не могут иметь порядок выше 1-го для жестких задач, поэтому они эффективны только при низкой задаваемой точности. Более эффективные методы 2-го порядка получим, задав $s = 4$. Для эффективного решения жестких задач следует задать $\beta = 1$, но для нежестких задач оптимальное значение β равно 2/3 (тогда метод имеет 3-й порядок). Поэтому имеет смысл настраивать в процессе решения не только заключительную формулу интегрирования, но и параметр β, что позволяет наиболее эффективно решать не только жесткие, но и нежесткие задачи.

Характерной особенностью многих жестких задач является наличие в решении коротких пограничных участков (слоев) с быстрым изменением переменных. При прохождении этих участков все значения $h\lambda_i$ невелики, поэтому задача внутри такого участка не является жесткой. Таким образом, все решение можно разбить на “медленные” жесткие и “быстрые” нежесткие участки. На нежестких участках изменения переменных наиболее значительны, поэтому они вносят значительный вклад в ошибку численного решения. Это является еще одной причиной для построения методов, имеющих повышенный порядок сходимости при решении нежестких задач.

Построим такой метод на основе стадий (3.1) при $s = 4$. Параметры α и β задаем исходя из полученных на предыдущем шаге оценок $\tilde{z}_i$, при этом $\beta = 1 - \alpha$, а α принимаем в виде (3.2), где $\bar{\alpha} = 1/3$. Такие значения обеспечивают эффективное решение как жестких, так и нежестких задач. Шаг интегрирования выполняем по формуле

$$\mathbf{y}_1 = \mathbf{y}_0 + h\big(\mathbf{f}_0 + \mathbf{u}_2/2 + \mathbf{d}_3\mathbf{u}_3\big), \tag{5.1}$$

где $\mathbf{u}_i = (\mathbf{F}_i - \mathbf{F}_{i-1})/(\beta\alpha^{i-2})$, а коэффициенты вектора $\mathbf{d}_3$ вычисляем по формулам (2.2), (2.5), (2.7).

При реализации алгоритмов с автоматическим выбором размера шага используют вложенную пару формул (методов) вычисления $\mathbf{y}_1$ и $\hat{\mathbf{y}}_1$, позволяющую получить оценку локальной ошибки как норму вектора $\mathbf{y}_1 - \hat{\mathbf{y}}_1$. В [7], [8] при $\beta = 1$, $s = 4$ использовалась вложенная формула вида

$$\hat{\mathbf{y}}_1 = \mathbf{y}_0 + h(\mathbf{f}_0 + \hat{\mathbf{d}}_2\mathbf{u}_2), \tag{5.2}$$

где компоненты вектора $\hat{\mathbf{d}}_2$ вычисляем согласно (2.5) с использованием скалярной функции устойчивости вложенного метода $\hat{Q}(z)$. Но если $\beta < 1$ и внутри интервала $[t_0 + \beta h, t_0 + h]$ происходит быстрое изменение правой части, то на значениях (5.1) и (5.2) это никак не сказывается, вследствие чего снижается точность решения (см. пример в разд. 2.4 книги [7]). Поэтому при $\beta < 1$ оценка ошибки на основе (5.2) ненадежна, и для построения формулы вычисления $\hat{\mathbf{y}}_1$ следует использовать также значение $\mathbf{f}_1 = \mathbf{f}(t_0 + h, \mathbf{y}_1)$. Это же значение является результатом выполнения первой стадии следующего шага, поэтому такие вложенные пары называют FSAL (First Same As Last, см. [23]).

Для FSAL-пары принимаем

$$\hat{\mathbf{y}}_1 = \mathbf{y}_0 + h(\mathbf{f}_0 + \hat{\mathbf{d}}_2\mathbf{u}_2 + \hat{\mathbf{d}}_3\mathbf{u}_3 + \hat{\mathbf{d}}_4\mathbf{v}_4), \quad \mathbf{v}_4 = \mathbf{f}_1 - \mathbf{f}_0 - \mathbf{u}_2 - \mathbf{u}_3/2. \tag{5.3}$$

Вычисление очередной компоненты векторов $\hat{\mathbf{d}}_2$, $\hat{\mathbf{d}}_3$ и $\hat{\mathbf{d}}_4$ выполняем, используя оценку $\tilde{z}$ по этой компоненте:

$$\hat{d}_2 = (1-\gamma-g)\gamma + a + g(1-g), \quad \hat{d}_3 = ((1-\gamma-g)\gamma + a)g + a\gamma,$$
$$\hat{d}_4 = ag(2+4\gamma(1+\gamma)), \quad a = g\left(g-\frac{7}{9}\right)+\frac{53}{162}, \quad \gamma = \min\left(\frac{2}{9}, \left|\tilde{z}^{-1}\right|\right). \tag{5.4}$$

Эти значения выбраны с учетом условия 2-го порядка формулы (5.3) для нежестких компонент и 1-го порядка для жестких компонент. Кроме этого, функция устойчивости $\hat{R}(z) = 1 + z + \hat{d}_2 z^2 + \hat{d}_3 z^3 + \hat{d}_4 d_3 z^4$ удовлетворяет условию $\hat{R}(-1/\gamma) \approx 0$, а параметр g остается свободным (мы принимаем $g = 1/8$). Метод, основанный на формулах (5.1), (5.3), (5.4), обозначим через ARK32.

Рассмотрим теперь метод с коррекцией жестких компонент, которую выполняем после вычислений по формулам (5.1), (5.3), (5.4). Если для i-й компоненты $\tilde{z}_i < -4.5$, то пересчитываем эту компоненту согласно формулам

$$y_{1\text{cor}} = y_1 + \delta_3 u_3 + \delta_4 v_4, \quad \delta_3 = \gamma(1/2 - \gamma(2-3\gamma)),$$
$$\delta_4 = \delta_3(2+4\gamma(1+\gamma)), \quad \gamma = \left|\tilde{z}_i^{-1}\right|,$$

где y_1, u_3, v_4 суть i-е компоненты векторов $\mathbf{y}_1, \mathbf{u}_3, \mathbf{v}_4$. Метод с коррекцией жестких компонент обозначим через ARK32c. При решении некоторых жестких задач этот метод значительно более эффектен, чем ARK32.

Чтобы адаптивные методы были эффективными, необходимо ограничить рост внутренних функций устойчивости. В методах, основанных на стадиях (3.1), это осуществляется с помощью настройки параметра α. Однако реальный порядок таких методов для жестких задач не может быть выше 2-го. Построить эффективный метод, имеющий при решении жестких задач 3-й порядок, удалось только на основе явного метода, имеющего 2-й псевдостадийный порядок (см. [7]). Формулы явного адаптивного метода 3-го порядка со стабилизированной 1-й стадией приведены в [7], [8]. Обозначим этот метод через ARK3s.

6. ЧИСЛЕННЫЕ ЭКСПЕРИМЕНТЫ

Для экспериментов с переменным размером шага были выбраны 5 задач из [1], основные характеристики которых приведены в табл. 1. Для управления размером шага используем стандартную процедуру (см. [1], [7]). Допустимую ошибку обозначим через *Tol*. Для всех задач задаем допустимую относительную ошибку $Rtol = Tol$, а допустимую абсолютную ошибку принимаем в виде $Atol = Tol$ для задач VDPOL и OREGO, $Atol = 10^{-6} \times Tol$ для ROBER, $Atol = 10^{-4} \times Tol$ для HIRES и $Atol = 10^{-2} \times Tol$ для CUSP. Вычислительные затраты оцениваем числом вычислений правой части *Nf*, а в качестве оценки точности решения, как и в [24], используем значение

$$scd = -\lg\left(\max_i\left(|y_i - \tilde{y}_i|/|y_i|\right)\right),$$

где y_i – точное, а $\tilde{y}_i$ – численное решение i-й компоненты в конце интервала. Для решения, наряду с рассмотренными в [7], [8] методами ARK21s, ARK2s и ARK3s, применялись новые методы ARK2c, ARK32 и ARK32c. Метод ARK2c отличается от ARK32c только значением $\beta = 1$.

Результаты для трех значений *Tol* приведены в табл. 6. Обсудим сначала результаты решения задачи VDPOL, которая задается уравнениями

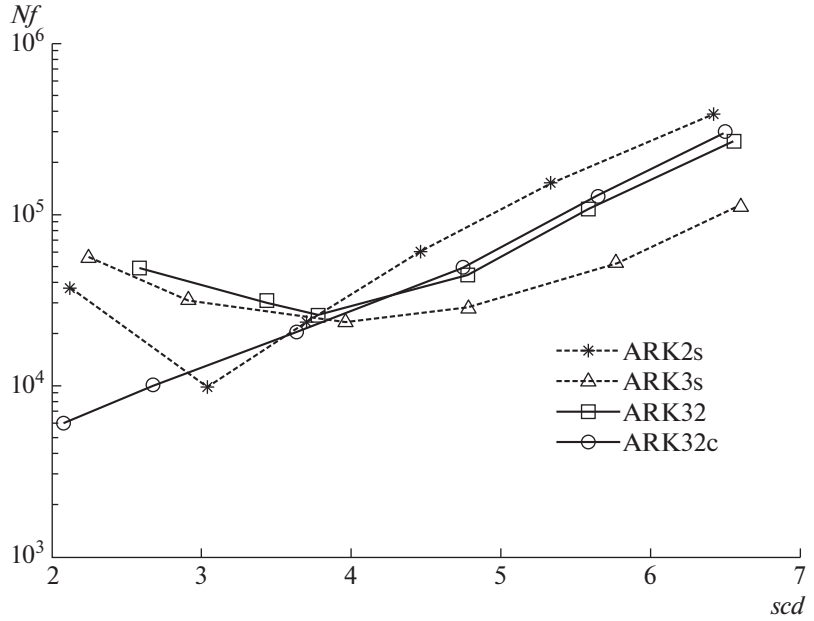
$$y_1' = y_2, \quad y_2' = \mu((1-y_1^2)y_2 - y_1), \quad y_1(0) = 2, \quad y_2(0) = 0, \quad 0 \le t \le 2, \tag{6.1}$$

при $\mu = 10^6$. Метод ARK21s эффективен только при низкой задаваемой точности, но он позволяет решать очень жесткие задачи, например, задача (6.1) была успешно решена при $\mu = 10^{46}$. Метод ARK2c заметно уступает методу ARK32c по точности, что объясняется его меньшим порядком на нежестких участках. При решении других задач методы ARK21s и ARK2c также оказались

Таблица 6. Результаты решения тестовых задач

Задача	Метод	$Tol = 10^{-2}$		$Tol = 10^{-3}$		$Tol = 10^{-4}$	
		scd	*Nf*	*scd*	*Nf*	*scd*	*Nf*
VDPOL	ARK21s	2.16	2515	3.27	16183	5.02	103895
	ARK2s	2.25	823	4.14	1708	4.06	4060
	ARK3s	2.31	1181	3.06	1276	3.56	2276
	ARK2c	1.12	1120	2.09	2071	2.64	4207
	ARK32	2.69	1705	2.99	2437	4.15	4069
	ARK32c	2.44	1093	3.11	2029	4.13	4110
ROBER	ARK2s	3.05	32383	3.39	895	4.37	2629
	ARK3s	2.96	41971	2.61	21156	4.74	10031
	ARK32	4.38	28377	6.23	18641	5.76	8221
	ARK32c	3.84	925	4.17	1394	4.47	2330
OREGO	ARK2s	1.06	1984	2.19	4360	3.28	11275
	ARK3s	1.71	2451	2.18	3036	3.19	4971
	ARK32	1.70	3905	2.47	4649	2.67	8109
	ARK32c	0.95	1870	1.67	3598	2.92	8883
HIRES	ARK2s	1.35	1129	2.05	1489	2.57	2338
	ARK3s	1.22	2216	3.06	2341	3.78	2666
	ARK32	1.01	1765	1.37	1725	2.22	2381
	ARK32c	0.73	1344	1.29	1652	2.71	2293
CUSP	ARK2s	2.88	565	3.46	1306	4.26	3091
	ARK3s	3.05	7086	3.66	3611	4.53	3331
	ARK32	3.16	13349	4.16	3733	4.11	2685
	ARK32c	2.42	679	3.18	1185	3.91	2826

менее эффективными по сравнению с другими методами, поэтому мы исключаем их из дальнейшего рассмотрения. Некоторые результаты методов ARK21, ARK21s и ARK21c приведены в [7] (эти методы можно рекомендовать для решения очень жестких задач с низкой точностью).



Фиг. 4.

Обратим теперь внимание на результаты методов ARK2s, ARK3s и ARK32 при решении задачи ROBER. При $Tol = 10^{-2}$ вычислительные затраты Nf намного больше, чем при меньших значениях Tol. Объясняется это тем, что при увеличении размера шага возникает неустойчивость численного решения, которая приводит к резким колебаниям величины шага и увеличению числа шагов. Аналогичное явление проявляется и при решении задачи CUSP методами ARK3s и ARK32 (хотя задача CUSP значительно менее жесткая, чем задачи VDPOL и OREGO, при решении которых этот эффект отсутствует).

Для более наглядного сравнения методов мы объединили результаты решения всех пяти задач и приводим на фиг. 4 зависимости общих затрат на их решение от усредненного для этих задач значения scd при $Tol = 10^{-i}$, $i = 2, \ldots, 7$. При $Tol = 10^{-4}$ все методы показывают примерно одинаковые результаты. При повышении точности ($Tol < 10^{-4}$) преимущество имеет метод более высокого порядка ARK3s, а при $Tol > 10^{-4}$ метод ARK32c превосходит другие методы.

Анализ приведенных результатов позволяет сделать вывод, что наиболее эффективным явным адаптивным методом при решении жестких задач с умеренной точностью является ARK32c, а для расчетов с повышенной точностью можно использовать метод ARK3s.

СПИСОК ЛИТЕРАТУРЫ

1. *Хайрер Э., Ваннер Г.* Решение обыкновенных дифференциальных уравнений. Жесткие и дифференциально-алгебраические задачи. М.: Мир, 1999.
2. *Curtiss C.W., Hirschelder J.O.* Integration of stiff equations // Proc. Nat. Acad. Sci. USA. 1952. V. 38. P. 235–243.
3. *Лебедев В.И.* Как решать явными методами жесткие системы дифференциальных уравнений // Вычисл. процессы и системы. М.: Наука, 1991. Вып. 8. С. 237–291.
4. *Verwer J.G.* Explicit Runge–Kutta methods for parabolic partial differential equations // Appl. Numer. Math. 1996. V. 22. № 1–3. P. 359–379.
5. *Новиков Е.А.* Явные методы для жестких систем. Новосибирск: Наука, 1997.
6. *Скворцов Л.М.* Явные стабилизированные методы Рунге–Кутты // Ж. вычисл. матем. и матем. физ. 2011. Т. 51. № 7. С. 1236–1250.
7. *Скворцов Л.М.* Численное решение обыкновенных дифференциальных и дифференциально-алгебраических уравнений. М: ДМК Пресс, 2018.
8. *Скворцов Л.М.* Явные адаптивные методы Рунге–Кутты для жестких и колебательных задач // Ж. вычисл. матем. и матем. физ. 2011. Т. 51. № 8. С. 1434–1448.
9. *Скворцов Л.М.* Явные адаптивные методы Рунге–Кутты // Матем. моделирование. 2011. Т. 23. № 7. С. 73–87.
10. *Fowler M.E., Warten R.M.* A numerical integration technique for ordinary differential equations with widely separated eigenvalues // IBM J. Research and Development. 1967. V. 11. № 5. P. 537–543.
11. *Lambert J.D.* Nonlinear methods for stiff systems of ordinary differential equations // Lect. Notes in Math. 1974. V. 363. P. 75–88.
12. *Wambecq A.* Rational Runge–Kutta methods for solving systems of ordinary differential equations // Computing. 1978. V. 20. № 4. P. 333–342.
13. *Бобков В.В.* Новые явные А-устойчивые методы численного решения дифференциальных уравнений // Дифференц. ур-ния. 1978. Т. 14. № 12. С. 2249–2251.
14. *Заворин А.Н.* Применение нелинейных методов для расчета переходных процессов в электрических цепях // Изв. вузов. Радиоэлектроника. 1983. Т. 26. № 3. С. 35–41.
15. *Скворцов Л.М.* Адаптивные методы численного интегрирования в задачах моделирования динамических систем // Изв. РАН. Теория и системы управления. 1999. № 4. С. 72–78.
16. *Скворцов Л.М.* Явные адаптивные методы численного решения жестких систем // Матем. моделирование. 2000. Т. 12. № 12. С. 97–107.
17. *Скворцов Л.М.* Явный многошаговый метод численного решения жестких дифференциальных уравнений // Ж. вычисл. матем. и матем. физ. 2007. Т. 47. № 6. С. 959–967.
18. *Козлов О.С., Скворцов Л.М., Ходаковский В.В.* Решение дифференциальных и дифференциально-алгебраических уравнений в программном комплексе “МВТУ”. 2005. URL: http://model.exponenta.ru/mvtu/20051121.html.

19. *Козлов О.С., Скворцов Л.М.* Программный комплекс “МВТУ” в научных исследованиях и прикладных разработках // Матем. моделирование. 2015. Т. 27. № 11. С. 32–46.
20. *Карташов Б.А., Шабаев Е.А., Козлов О.С., Щекатуров А.М.* Среда динамического моделирования технических систем SimInTech. М.: ДМК Пресс, 2017.
21. *Деккер К., Вервер Я.* Устойчивость методов Рунге–Кутты для жестких нелинейных дифференциальных уравнений. М.: Мир, 1988.
22. *Shampine L.F., Reichelt M.W.* The MATLAB ODE suite // SIAM J. Sci. Comput. 1997. V. 18. № 1. P. 1–22.
23. *Bogacki P., Shampine L.F.* A 3(2) pair of Runge–Kutta formulas // Appl. Math. Lett. 1989. V. 2. № 4. P. 321–325.
24. *Mazzia F., Magherini C.* Test set for initial value problem solvers. Release 2.4. 2008. URL: http://pitagora.dm.uniba.it/~testset/report/testset.pdf.