УДК 519.62 doi 10.26089/NumMet.v21r107

ЭФФЕКТИВНАЯ ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ЧИСЛЕННЫХ МЕТОДОВ РЕШЕНИЯ ЖЕСТКИХ СИСТЕМ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ С ЗАПАЗДЫВАЮЩИМ АРГУМЕНТОМ

Д. А. Желтков[1], Р. М. Третьякова[2], В. В. Желткова[3], Г. А. Бочаров[4]

Системы уравнений с запаздываниями широко применяются в различных областях современного математического моделирования. В ходе разработки структуры математической модели и идентификации ее параметров приходится многократно решать задачу Коши для подобных систем. В случае высокой размерности системы, а также при условии жесткости задачи численное решение уравнений с запаздываниями может требовать значительных вычислительных и временны́х затрат. Таким образом, разработка и реализация эффективных алгоритмов численного решения различных классов уравнений с запаздывающими аргументами является актуальной задачей. В настоящей статье представлена модифицированная версия программного комплекса DIFSUBDEL, в которой реализованы методы численного решения дифференциальных уравнений с запаздываниями на основе линейных многошаговых методов. Переработанная версия разработана с применением принципов структурного программирования и является значительно более удобной в эксплуатации, чем исходная, а также обладает свойством потокобезопасности, что позволяет использовать комплекс в качестве блока в системах, основанных на технологиях параллельного программирования с общей памятью. Был проведен сравнительный анализ производительности переработанной системы DIFSUBDEL с другими существующими программными реализациями численных методов решения дифференциальных уравнений с запаздыванием и показана ее эффективность.

Ключевые слова: численные методы, дифференциальные уравнения с запаздываниями, жесткие системы, линейные многошаговые методы.

1. Введение. Системы дифференциальных уравнений с запаздывающим аргументом (ДУЗА) широко используются для построения математических моделей в таких областях, как физика, биология, экология, иммунология [1, 2]. Так, в биологических моделях запаздывание может отражать длительность процессов, не описываемых напрямую, однако занимающих некоторое время, например, транспортировку веществ, некоторые стадии клеточного цикла при дифференцировке и пролиферации клеток. По сравнению с моделями на основе обыкновенных дифференциальных уравнений (ОДУ), модели, основанные на ДУЗА, позволяют описать более широкий класс процессов, обладают более богатой динамикой [3, 4], а также могут обеспечить лучшую согласованность с экспериментальными данными по кинетике рассматриваемых переменных. Для решения задачи Коши для системы ДУЗА предложено и описано достаточно большое количество численных методов (например, обзоры можно найти в работах [5, 6]). Однако количество разработанных программных реализаций (например, [7–10]) существенно меньше, чем для систем ОДУ. При этом большинство программных реализаций сложны в понимании и модификации, так как написаны на языке FORTRAN-77, в котором отсутствовали многие возможности, доступные в более новых стандартах. Данная проблема рассматривалась ранее для некоторых программных комплексов. Так, были созданы дружественные к пользователю версии солвера для уравнений с запаздываниями DKLAG6, а также для решения краевых задач для ОДУ [11, 12]. При этом авторы использовали FORTRAN-90/95, а также руководствовались опытом написания программ на MATLAB [24].

В настоящей работе решена задача развития программного комплекса DIFSUBDEL, предложенного в 1986 г. Г. А. Бочаровым и А. А. Романюхой [13, 14]. Этот комплекс позволяет эффективно решать

[1] Институт вычислительной математики им. Г. И. Марчука РАН, ул. Губкина, д. 8, 119333, г. Москва; мл. науч. сотр., e-mail: dmitry.zheltkov@gmail.com
[2] Институт вычислительной математики им. Г. И. Марчука РАН, ул. Губкина, д. 8, 119333, г. Москва; мл. науч. сотр., e-mail: trrufina@yandex.ru
[3] Институт вычислительной математики им. Г. И. Марчука РАН, ул. Губкина, д. 8, 119333, г. Москва; мл. науч. сотр., e-mail: valeryaaziattseva@yandex.ru
[4] Институт вычислительной математики им. Г. И. Марчука РАН, ул. Губкина, д. 8, 119333, г. Москва; вед. науч. сотр., e-mail: bocharov@m.inm.ras.ru

задачу Коши для систем ДУЗА, в том числе при условии жесткости. Нашей целью была переработка системы DIFSUBDEL с использованием более современной версии языка FORTRAN-90 для улучшения читаемости исходного кода, упрощения его модификаций, а также выполнения условий потокобезопасности для возможности применения технологий параллельного программирования, использующих общую память. Было проведено тестирование переработанного программного комплекса, а также сравнение с другими существующими программными реализациями численных методов решения систем ДУЗА на примере системы уравнений, описывающих модель гепатита-B [15]. Было показано, что разработанный программный комплекс позволяет эффективно решать задачу Коши для систем уравнений с запаздывающим аргументом, в том числе при условии жесткости задачи. Модернизированная версия комплекса DIFSUBDEL выложена в открытый репозиторий Bitbucket [16].

2. Численное решение задачи Коши для ДУЗА. Общий принцип построения численных методов решения систем ДУЗА на основе адаптации алгоритмов для ОДУ основан на использовании метода шагов, предложенного в [17], и описан в [1]. Рассмотрим задачу Коши для неавтономной системы ОДУ:

$$\mathbf{y}'(t) = \mathbf{g}(t, \mathbf{y}(t)), \quad t \geqslant t_0, \quad \mathbf{y}(t_0) = \mathbf{y}_0. \tag{1}$$

С помощью конечно-разностного метода и реализующего его численного алгоритма вычисляется приближенное значение решения $\tilde{\mathbf{y}}(t_i)$ в последовательности точек $t_0 < t_1 < \ldots < t_N$, определяемой алгоритмом, например по значению локальной ошибки. Дополнительно, используя алгоритм интерполяции, можно приближенно вычислить значения решения $\tilde{\mathbf{y}}(t_i)$ в произвольный момент времени внутри отрезка $[t_0, t_N]$. Такие численные методы для решения ОДУ могут быть модифицированы для вычисления приближенного решения систем ДУЗА. Рассмотрим задачу Коши для системы ДУЗА с параметром $\mathbf{p}$:

$$\mathbf{y}'(t) = \mathbf{f}(t, \mathbf{y}(t), \mathbf{y}(t-\tau_*), \mathbf{p}), \quad t \geqslant t_0, \quad \mathbf{y}(t) = \psi(t, \mathbf{p}), \quad t \in [t_0 - \tau_*, t_0]. \tag{2}$$

Предположим, что $\tau_* > 0$. Для того чтобы найти решение $\mathbf{y}(t)$ при $t \geqslant t_0$, требуется задать начальные значения ψ на интервале $t \in [t_0 - \tau_*, t_0]$. Наиболее естественный подход к численному решению системы (2) заключается в использовании метода шагов [17], т. е. ее замене на систему ОДУ на каждом отрезке $[n\tau_*, (n+1)\tau_*]$:

$$\mathbf{y}'(t) = \mathbf{f}(t, \mathbf{y}(t), \tilde{\mathbf{y}}(t-\tau_*), \mathbf{p}), \quad t \geqslant t_n, \quad t \in [n\tau_*, (n+1)\tau_*]. \tag{3}$$

При этом предполагается, что $\tilde{\mathbf{y}}(t-\tau_*)$ известно, так как мы вычисляем значения $\tilde{\mathbf{y}}$, $t \leqslant t_n$, путем интерполяции решения. В общем виде численный метод решения ДУЗА может быть представлен в виде комбинации двух ключевых компонент:

- Метод π_q для аппроксимации запаздывающих переменных с порядком q на некоторой сетке $\{t_i\}_{i=1}^N$, в узлах которой известно значение решения;
- Метод для численного решения ОДУ Φ_p p-го порядка для нахождения решения с шагом h_n (предполагаем, что $\tau_* > h_n$) в узлах сетки $t_{n+1} = t_n + h_n$, $n = 1, \ldots, N-1$.

Для построения эффективного метода численного решения ДУЗА с заданной точностью требуется контроль размера шага и согласованность π_q и Φ_p, так как некоторые свойства ДУЗА могут существенно повлиять на надежность и эффективность численного метода $(\pi_q, \Phi(p))$. В общем случае решение системы (2) не является гладким, имеются разрывы первого рода i-х производных в точках $t_i = t_0 + i\tau_*$, $i \in N^+$. Для того чтобы адаптированный метод (π_q, Φ_p) сходился с порядком p, требуется выполнение условия $q \geqslant p-1$. Сохранение в алгоритме схемы контроля величины шага требует, чтобы $q \geqslant p$. Данные вопросы были систематически исследованы в [15].

2.1. Существующие программные реализации. Большинство численных методов решения систем ДУЗА реализовано на языке FORTRAN. Комплекс DKLAG6 [18] основан на использовании семейства методов Рунге–Кутты–Сарафяна [19] шестого порядка для решения функциональных дифференциальных уравнений с запаздываниями, зависящими от состояния. В системе DELSOL [8] используется модификация линейного многошагового метода Адамса, реализованного по схеме предиктор–корректор $P_kEC_{k+1}E$. Помимо описанных комплексов, существуют и другие (например, ARCHI [20], DMRODE [9], DDESTRIDE [21], SNDDELM [22] и dde23 [7]).

Среди существующих методов следует отметить те, для которых реализованы интерфейсы, позволяющие осуществлять их вызов из высокоуровневых языков. Так, алгоритм dde23 [7, 23] системы MATLAB [24] является расширением солвера ode23 (основанном на явном методе Рунге–Кутты 2–3 порядков в реализации Богаки–Шампейн (Bogacki–Shampine)) для систем нежестких ОДУ с использованием метода шагов. Система Radar5 реализована на FORTRAN-90 на основе неявного метода Рунге–Кутты 5-го порядка

(Radau IIA) и может быть использована для решения жестких задач. Для Radar5 реализован модуль-оболочка radar5 на языке Python [10].

В данной работе мы рассматривали комплекс DIFSUBDEL, предложенный Г. А. Бочаровым и А. А. Романюхой в 1986 г. В этом комплексе реализованы два подкласса линейных многошаговых методов: метод Адамса–Бэшворта–Мултона (ABM) переменного шага и переменного порядка (от 1 до 7) и методы Гира (BDF) на основе формул дифференцирования назад переменного шага и переменного порядка (от 1 до 6), реализованные по схеме $P(EC)^M$, $M \leqslant 3$. Теоретические аспекты предложенных методов, вопросы аппроксимации, сходимости и устойчивости рассмотрены в [13], а детали исходной реализации алгоритма и текст программы — в [14]. Комплекс DIFSUBDEL позволяет эффективно решать задачу Коши для систем ДУЗА, в том числе при условии жесткости.

3. Обновленный комплекс DIFSUBDEL. Изначально программный комплекс DIFSUBDEL был реализован на FORTRAN-77, в связи с чем обладает рядом недостатков, значительно затрудняющих работу с исходным кодом и реализацию новых математических моделей на его основе. Так, FORTRAN-77 предполагает фиксированную форму записи, что является непривычным для большинства современных пользователей. Присутствие неявных типов данных приводит к появлению ошибок, не обнаруживаемых на этапе компиляции программы. Широкое использование оператора GOTO значительно затрудняет чтение, понимание и редактирование исходного кода. COMMON-блоки тоже создают трудности при редактировании программ, а также не являются потокобезопасными, из-за чего невозможна параллелизация с использованием общей памяти.

Нашей целью была переработка программного комплекса на основе более современного стандарта FORTRAN-90 с целью улучшения читаемости исходного кода, упрощения его изменения и возможности дальнейшего использования. Важной задачей было сделать программу потокобезопасной, в результате чего становится возможным параллельный запуск нескольких экземпляров DIFSUBDEL с использованием таких технологий, как OpenMP или pThread (эта возможность особенно полезна, например, при идентификации параметров модели, когда приходится многократно решать обратную задачу). При переработке программного комплекса мы придерживались следующих принципов.

- В соответствии с парадигмой структурного программирования [25] были исключены операторы условного и безусловного перехода (GOTO, SELECT). Вместо них использовались базовые управляющие конструкции (последовательность, ветвление, цикл).
- В соответствии с принципом модульности каждая подпрограмма была вынесена в отдельный файл.
- Повторяющиеся COMMON-блоки были заменены структурами, в результате чего программа стала потокобезопасной. Более того, данная модификация значительно уменьшила вероятность ошибок при изменении данных, представленных ранее в COMMON-блоках.
- В модифицированной версии программного комплекса мы отказались от неявной типизации.

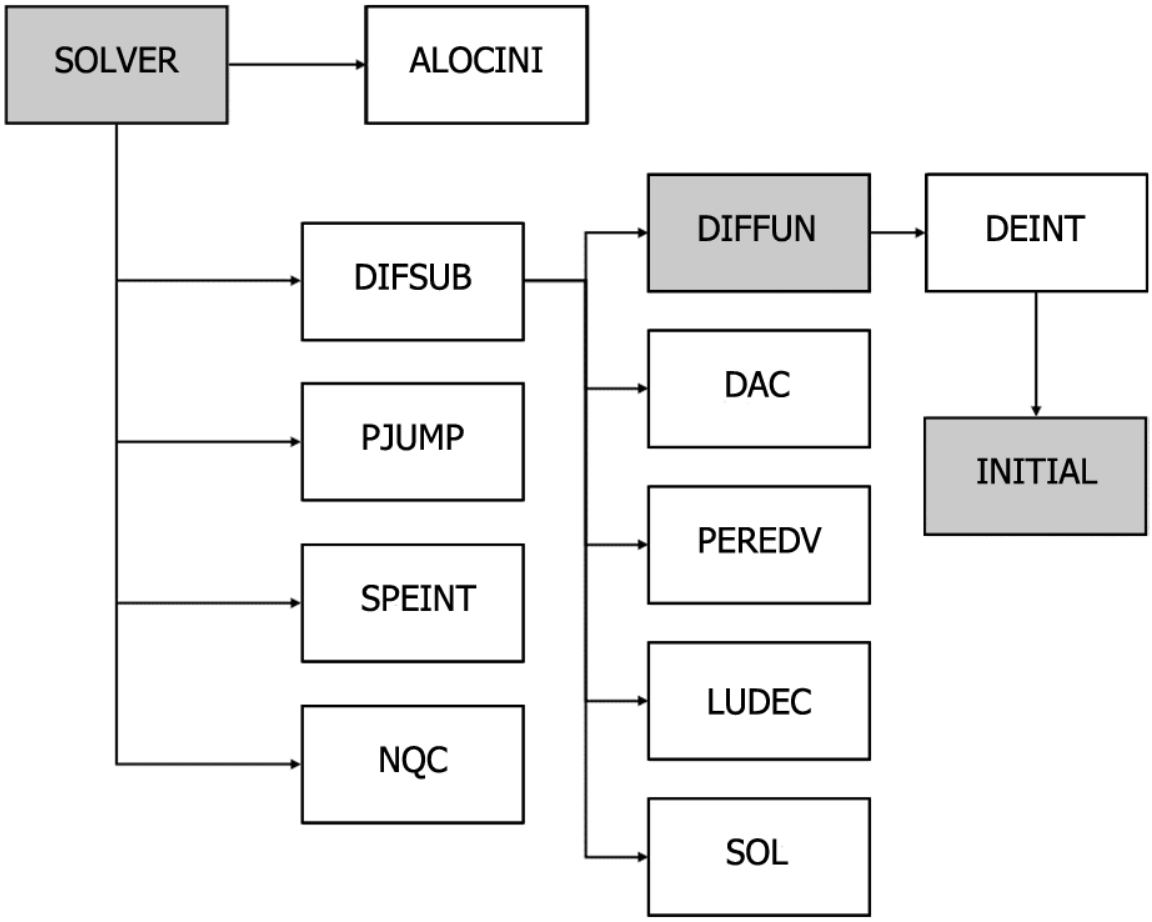
Помимо этого, для удобства использования был упрощен ввод пользовательских данных, а также реализован интерфейс вызова из языка C++.

3.1. Структура обновленного программного комплекса. Блок-схема обновленного комплекса программ DIFSUBDEL (см. рисунок) иллюстрирует взаимосвязь между процедурами. Процедуры программного комплекса описаны в табл. 1, структуры данных — в табл. 2–6. Процедуры и переменные, задаваемые пользователем, на схеме и в таблицах помечены серым цветом. В исходном коде такие процедуры и переменные помечены комментарием ***SET BY USER***.

В программе используется модуль COMMONS, разработанный с целью избавления от COMMON-блоков и уменьшения количества параметров, передаваемых в процедуры. Модуль содержит структуры TPROBLEM (см. табл. 2), TDELAY (см. табл. 3), TINTEG (см. табл. 4), TWORK (см. табл. 5), TDISCPT(см. табл. 6). В программном комплексе создается по одной переменной каждого из описанных типов.

3.2. Результаты тестирования обновленного программного комплекса. Для валидации модифицированного программного комплекса были проведены расчеты на девяти тестовых задачах, описанных в [14].

Результаты, полученные с помощью модифицированного программного комплекса, совпадают с результатами расчетов, приведенных в [14]. Анализ полученных результатов свидетельствует о том, что алгоритм DIFSUBDEL позволяет достаточно надежно и эффективно решать некоторые классы жестких систем ДУЗА.



Структура обновленного комплекса DIFSUBDEL
Серым цветом выделены функции, изменяемые пользователем

Был проведен сравнительный анализ обновленного программного комплекса DIFSUBDEL с другими существующими реализациями численного решения систем уравнений с запаздываниями. Для анализа использовалась математическая модель вирусного гепатита-B, описанная в [15]. Уравнения, переменные и параметры модели приведены в исходной работе [15]. Модель представлена в виде системы из 10 нелинейных ДУЗА, обладающей свойством жесткости, и имеет 5 запаздываний.

Для сравнения использовался комплекс Radar5 [10], в котором данная модель уже была реализована в качестве одного из тестовых примеров, а также метод dde23, представленный в системе MATLAB. Результаты тестирования трех программных комплексов приведены в табл. 7. Для тестирования использовались следующие параметры методов:

- Относительная ошибка RTOL варьировалась от 10^{-3} до 10^{-11} ;
- Значение абсолютной точности задавалось равным $\mathrm{RTOL} * 10^{-20}$;
- Стартовое значение шага интегрирования задавалось равным 10^{-6}.

Для каждого метода рассматривались величины **NF** (количество вычислений правой части ДУЗА) и **NH** (количество шагов метода) для требуемого значения относительной ошибки. Результаты приведены в табл. 7. Как и ожидалось, метод dde23 требует большого количества вычислений, так как не предназначен для решения жестких задач. Методы DIFSUBDEL и Radar5 показывают схожие результаты. Практически во всех проведенных тестах при интегрировании системы с помощью DIFSUBDEL потребовалось наименьшее количество вычислений правой части ДУЗА. При высокой требуемой относительной точности количество шагов метода DIFSUBDEL меньше, чем у Radar5. Таким образом, усовершенствованный программный комплекс DIFSUBDEL позволяет эффективно численно решать жесткие системы ДУЗА.

Таблица 1. Процедуры комплекса DIFSUBDEL

Процедуры комплекса DIFSUBDEL.	
SOLVER	основная процедура программного комплекса; Пользователь изменяет только ту часть процедуры, которая отвечает за инициализацию и вывод результатов. Инициализация происходит в следующем порядке: – задаются размерности системы и рабочих массивов (N,NDEL, NYD, KPR, L, MXJMP); – вводится информация о связи запаздываний и переменных (массив MD); – вызывается процедура ALOCINI, распределяющая память для всех динамических массивов программы; – открывается файл для записи результатов, и задается формат записи – задается начальное время T0 и начальные значения Y –устанавливаются параметры метода и интегратора (GROUND, MF, MAXDER, EPS, NCUT, HMIN, HMAX0) –задается шаг печати TRPINT (опционально) Далее программа производит инициализацию части переменных, находит точки разрыва производных решения при помощи процедуры PJUMP, после чего выполняет цикл интегрирования системы ДУЗА. Внутри цикла, а также после выхода из него пользователь может организовать вывод и запись результатов расчетов.
DIFFUN	процедура, вычисляющая правые части системы ДУЗА в момент времени T. Сначала в процедуре вычисляются значения запаздывающих переменных при помощи процедуры DEINT. Результат записывается в массив VALUE_1. Далее пользователем задаются функции правых частей ДУЗА: YP(I) $= f_i$. В правых частях могут присутствовать переменные Y(1,I) $= y_i$, запаздывающие переменные VALUE_1(I,J) $= y_i(t - \tau_j)$, а также параметры RPAR(I);
INITIAL	процедура задает начальные значения переменных на интервале $[t_0 - \tau_{max}, t_0]$. Процедура возвращает значение только одной переменной YD1 $= y_i$ по индексу IY $= i$, и времени T;
ALOCINI	производит распределение памяти динамических массивов и инициализирует массивы IYD, IVDS, ILENG;
PJUMP	вычисляет точки разрывов производных решения.
SPEINT	на каждом шаге интегрирования определяет TOUT – время выхода из процедуры DIFSUB и максимальный размер шага интегрирования HMAX.
NQC	на каждом шаге интегрирования осуществляет контроль за порядком метода интегрирования NQ $= p_n$.
DIFSUB	интегратор.
LUDEC, SOL	осуществляют LU-разложение и решение системы СЛАУ.
DEINT	аппроксимирует запаздывающую переменную с индексом IDEL, вызывает процедуру INITIAL.
DAC	производит запись данных в массивы TDEL, YDEL, NQDEL, NADEL.

Таблица 2. Структура TPROBLEM

TPROBLEM: содержит переменные, характеризующие систему ДУЗА.	
N	число переменных ДУЗА;
NDEL	число различных запаздываний τ;
NYD	число запаздывающих переменных;
KPR	число параметров системы ДУЗА;
RPAR	вектор параметров системы ДУЗА, размерности KPR;
DELAY	вектор запаздываний, размерности NDEL;
MD	массив размерности (N, NDEL), связывающий запаздывания и переменные, MD(i,j) $= 1$ означает наличие $y_i(t - \tau_j)$ в системе, MD(i,j) $= 0$ – отсутствие.
ILENG, IYD, IVDS	массивы, хранящие дополнительную информацию о запаздывающих переменных, заполняются процедурой ALOCINI.

Таблица 3. Структура TDELAY

TDELAY: содержит данные, необходимые для аппроксимации запаздывающих переменных	
TDEL, YDEL, NQDEL, NADEL	массивы, хранящие значения переменных t_n, $\bar{y}_n$, p_n и указатели;
L	L = (MXTDEL, MXYDEL), массив хранящий размерности TDEL, YDEL. Рекомендованное соотношение размеров: L(2) = 8*NYD*L(1). Если размер i-го массива недостаточен, программа завершает работу с сообщением «L(I) SIZE TOO SMALL».
ICAPAC, ISUCST	указатели в массивах TDEL, YDEL;
NLAST, NLASTW	указатели на элементы массива TDEL, использованные на предыдущих шагах аппроксимации;
IDMAX	индекс максимального запаздывания;
VALUE_1	массив размерности (N, NDEL), VALUE_1(i, j) = $y_i(t - \tau_j)$, если такое запаздывание присутствует в правой части ДУЗА, VALUE_1(i, j) = 0 иначе (используется для задания правых частей в процедуре DIFFUN).

Таблица 4. Структура TINTEG

TINTEG: содержит информацию о методе интегрирования и рабочие массивы интегратора	
MF	индекс метода интегрирования (0 (ABM) или 2 (BDF));
MAXDER	максимальный порядок метода (6 или 7);
NCUT	максимальное число последовательных уменьшений шага интегрирования
EPS	требуемая точность
GROUND	порог, до которого контролируется относительная ошибка решения
A	коэффициенты численной схемы
SAVER, YP1, YP2, YMAX, ERROR, PW, ALU, IP	рабочие массивы

Таблица 5. Структура TWORK

TWORK: содержит рабочие переменные интегратора.	
HOLD, HNEW, E, BND, EUP, EDWN, TOLD, ENQ1, ENQ2, ENQ3, IDOUB, IWEVAL, MTYP, N1,N2, N3, N4, N5, N6, NQ, NEWQ, NQOLD, DELMIN, K_W, INTCON,T0, INDJ	внутренние переменные интегратора
H_W	начальный шаг интегрирования, по умолчанию равен HMIN

Таблица 6. Структура TDISCPT

TDISCPT: содержит информацию о точках разрывов производных.	
MXJMP	максимальная размерность рабочих массивов TJUMP, TJMAXD
NJUMP	реальная размерность TJUMP, равная количеству разрывов
MAXD2	реальная размерность TJMAXD
TJUMP	массив уникальных размеров разрывов
TJMAXD	массив разрывов, соответствующих максимальному запаздыванию
U26,IDMIN,INTJ,INES	внутренние рабочие переменные

Таблица 7. Сравнение DIFSUBDEL с существующими программными реализациями

Метод	NF	NH	Относительная точность
DIFSUBDEL	1801	422	10^{-3}
Radar5	2563	318	
dde23	424405	141405	
DIFSUBDEL	2519	573	10^{-5}
Radar5	1894	217	
dde23	424567	141406	
DIFSUBDEL	3333	837	10^{-7}
Radar5	3658	469	
dde23	437989	144891	
DIFSUBDEL	4685	1288	10^{-9}
Radar5	7672	1015	
dde23	527593	170077	
DIFSUBDEL	6502	1926	10^{-11}
Radar5	16009	2200	
dde23	1060555	329417	

4. Заключение. В данной работе описана модификация программного комплекса DIFSUBDEL для решения систем дифференциальных уравнений с запаздывающим аргументом. Исходная система позволяла эффективно решать задачу Коши для ДУЗА, в том числе при условии жесткости задачи, однако обладала рядом недостатков с точки зрения практического использования, которые были исправлены в модифицированной версии. Переработанная система DIFSUBDEL реализована на FORTRAN-90 с использованием принципов структурного программирования и модульности. Новая версия является потокобезопасной и, таким образом, может использоваться в качестве модуля комплексов программ, основанных на технологиях параллельного программирования с использованием общей памяти. Проведено тестирование переработанной системы, показавшее корректность ее работы, а также выполнен сравнительный анализ производительности программного комплекса с другими существующими реализациями численного решения ДУЗА. В дальнейшем планируется разработка интерфейсов для вызова системы DIFSUBDEL из Python и MATLAB.

Работа выполнена при поддержке грантов РФФИ № 18–31–00356 (программная реализация) и РНФ № 18–11–00171 (тестирование и валидация).

СПИСОК ЛИТЕРАТУРЫ

1. *Baker C.T.H., Bocharov G.A., Rihan F.A.* A report on the use of delay differential equations in numerical modelling in the biosciences. Report MCCM 1360–1725. Manchester: Manchester Centre for Computational Mathematics, 1999.
2. *Bocharov G.A., Rihan F.A.* Numerical modelling in biosciences using delay differential equations // Journal of Computational and Applied Mathematics. 2000. **125**, N 1–2. 183-199.
3. *Kuang Y.* Delay differential equations: with applications in population dynamics. New York: Academic Press, 1993.
4. *Smith H.* An introduction to delay differential equations with applications to the life sciences. New York: Springer, 2011.
5. *Baker C.T.H., Paul C.A.H., Willé D.R.* A bibliography on the numerical solution of delay differential equations. Numer. Anal. Rept. No. 269. Manchester: Univ. of Manchester, 1995.
6. *Bellen A., Zennaro M.* Numerical methods for delay differential equations. Oxford: Oxford Univ. Press, 2013.
7. *Shampine L.F., Thompson S.* Solving delay differential equations with dde23. https://www.radford.edu/thompson/webddes/tutorial.pdf.
8. *Willé D.R., Baker C.T.H.* DELSOL — a numerical code for the solution of systems of delay-differential equations // Applied Numerical Mathematics. 1992. **9**, N 3–5. 223–234.
9. *Neves K.W.* Automatic integration of functional differential equations: an approach // ACM Trans. Math. Softw. 1975. **1**, N 4. 357–368.
10. Project radar5 0.1. https://pypi.org/project/radar5/.
11. *Thompson S., Shampine L.F.* A friendly Fortran DDE solver // Applied Numerical Mathematics. 2006. **56**, N 3–4. 503–516.
12. *Shampine L.F., Muir P.H., Xu H.* A user-friendly Fortran BVP solver // J. Numer. Anal. Ind. Appl. Math. 2006. **1**, N 2. 201–217.
13. *Бочаров Г.А., Романюха А.А.* Численное решение дифференциальных уравнений с запаздывающим аргументом на основе линейных многошаговых методов. Аппроксимация, сходимость и устойчивость. Препринт ОВМ АН СССР № 116. М., 1986.

14. *Бочаров Г.А., Романюха А.А.* Численное решение дифференциальных уравнений с запаздывающим аргументом на основе линейных многошаговых методов. Алгоритм и программа. Препринт ОВМ АН СССР № 117. М., 1986.
15. *Bocharov G., Marchuk G., Romanyukha A.* Numerical solution by LMMs of stiff delay differential systems modeling an immune response // Numerische Mathematik. 1996. **73**. 131–148.
16. DIFSUBDEL. `https://bitbucket.org/a\_valerya/difsubdel/src/master/`.
17. *Bellman R., Cooke K.L.* Differential-difference equations. New York: Academic Press, 1963.
18. *Corwin S.P., Sarafyan D., Thompson S.* DKLAG6: a code based on continuously imbedded sixth-order Runge–Kutta methods for the solution of state-dependent functional differential equations // Applied Numerical Mathematics. 1997. **24**, N 2–3. 319–330.
19. *Sarafyan D.* Approximate solution of ordinary differential equations and their systems through discrete and continuous embedded Runge–Kutta formulae and upgrading of their order // Computers and Mathematics with Applications. 1994. **28**, N 10–12. 353–384.
20. *Paul C.A.H.* A user-guide to Archi: an explicit Runge–Kutta code for solving delay and neutral differential equations and parameter estimation problems. Numerical Analysis Report No. 283 (Extended). Manchester: University of Manchester, 1997.
21. *Baker C.T.H., Butcher J.C., Paul C.A.H.* Experience of STRIDE applied to delay differential equations. Numerical Analysis Report No. 208. Manchester: University of Manchester, 1992.
22. *Lo E., Jackiewicz Z.* The algorithm SNDDELM for the numerical solution of systems of neutral delay differential equations // Delay Differential Equations with Applications in Population Dynamics. New York: Academic Press, 1993. 377–393.
23. *Shampine L.F., Thompson S.* Solving DDEs in Matlab // Applied Numerical Mathematics. 2001. **37**, N 4. 441–458.
24. *M*athWorks. https://www.mathworks.com/.
25. *Dahl O.J., Dijkstra E.W., Hoare C.A.R.* Structured programming. New York: Academic Press, 1972.

Поступила в редакцию
21.01.2020

An Efficient Software Implementation of Numerical Methods for Solving Stiff Systems of Delay Differential Equations

D. A. Zheltkov [1], R. M. Tretiakova [2] V. V. Zheltkova [3], and G. A. Bocharov [4]

[1] *Marchuk Institute of Numerical Mathematics, Russian Academy of Sciences; ulitsa Gubkina 8, Moscow, 119333, Russia; Junior Scientist, e-mail: dmitry.zheltkov@gmail.com*

[2] *Marchuk Institute of Numerical Mathematics, Russian Academy of Sciences; ulitsa Gubkina 8, Moscow, 119333, Russia; Junior Scientist, e-mail: trrufina@yandex.ru*

[3] *Marchuk Institute of Numerical Mathematics, Russian Academy of Sciences; ulitsa Gubkina 8, Moscow, 119333, Russia; Junior Scientist, e-mail: valeryaaziattseva@yandex.ru*

[4] *Marchuk Institute of Numerical Mathematics, Russian Academy of Sciences; ulitsa Gubkina 8, Moscow, 119333, Russia; Dr. Sci., Professor, Leading Scientist, e-mail: bocharov@m.inm.ras.ru*

Received January 21, 2020

Abstract: The systems of delay differential equation are widely used in modern mathematical modeling. The process of mathematical model development and identification requires the repeated solution of initial value problems for such systems. The numerical solution of delay differential equations may be computationally expensive, especially when the problem is stiff and high-dimensional. Therefore, it is important to develop and implement the efficient algorithms for the numerical solution of different classes of delay differential equations. In this paper, a new implementation of DIFSUBDEL program package for the numerical solution of delay differential equations based on linear multistep methods is discussed. The modified version is based on structured programming principles to make the program thread safe and user-friendly. The modified program performance is compared with the existing implementations of numerical methods for the solution of delay differential equations.

Keywords: numerical methods, delay differential equations, stiff systems, linear multistep methods.

References

1. C. T. H. Baker, G. A. Bocharov, and F. A. Rihan, *A Report on the Use of Delay Differential Equations in Numerical Modelling in the Biosciences*, Report MCCM 1360–1725 (Manchester Centre for Computational Mathematics, Manchester, 1999).

2. G. A. Bocharov and F. A. Rihan, "Numerical Modelling in Biosciences Using Delay Differential Equations," J. Comput. Appl. Math. **125** (1–2), 183–199 (2000).

3. Y. Kuang, *Delay Differential Equations with Applications in Population Dynamics* (Academic Press, New York, 1993).

4. H. Smith, *An Introduction to Delay Differential Equations with Applications to the Life Sciences* (Springer, New York, 2011).

5. C. T. H. Baker, C. A. H. Paul, and D. R. Willé, *A Bibliography on the Numerical Solution of Delay Differential Equations*, Numerical Analysis Report No. 269 (Univ. of Manchester, Manchester, 1995).

6. A. Bellen and M. Zennaro, *Numerical Methods for Delay Differential Equations* (Oxford Univ. Press, Oxford, 2013).

7. L. F. Shampine and S. Thompson, *Solving Delay Differential Equations with dde23*, https://www.radford.edu/thompson/webddes/tutorial.pdf. Cited February 9, 2020.

8. D. R. Willé and C. T. H. Baker, "DELSOL — a Numerical Code for the Solution of Systems of Delay-Differential Equations," Appl. Numer. Math. **9** (3–5), 223–234 (1992).

9. K. W. Neves, "Automatic Integration of Functional Differential Equations: An Approach," ACM Trans. Math. Softw. **1** (4), 357–368 (1975).

10. Project radar5 0.1. https://pypi.org/project/radar5. Cited February 9, 2020.

11. S. Thompson and L. F. Shampine, "A Friendly Fortran DDE Solver," Appl. Numer. Math. **56** (3–4), 503–516 (2006).

12. L. F. Shampine, P. H. Muir, and H. Xu, "A User-Friendly Fortran BVP Solver," J. Numer. Anal. Ind. Appl. Math. **1** (2), 201–217 (2006).

13. G. A. Bocharov and A. A. Romanyukha, *Numerical Solution of Delay-Differential Equations on the Basis of Linear Multistep Methods. Approximation, Convergence, and Stability*, Preprint No. 116 (Numer. Math. Department of USSR Academy of Sciences, Moscow, 1986).

14. G. A. Bocharov and A. A. Romanyukha, *Numerical Solution of Delay-Differential Equations on the Basis of Linear Multistep Methods. Algorithm and Program*, Preprint No. 117 (Numer. Math. Department of USSR Academy of Sciences, Moscow, 1986).

15. G. Bocharov, G. Marchuk, and A. Romanyukha, "Numerical Solution by LMMs of Stiff Delay Differential Systems Modeling an Immune Response," Numer. Math. **73**, 131–148 (1996).

16. DIFSUBDEL. `https://bitbucket.org/a_valerya/difsubdel/src/master/`. Cited February 9, 2020.

17. R. Bellman and K. L. Cooke, *Differential-Difference Equations* (Academic Press, New York, 1963).

18. S. P. Corwin, D. Sarafyan, and S. Thompson, "DKLAG6: A Code Based on Continuously Imbedded Sixth-Order Runge–Kutta Methods for the Solution of State-Dependent Functional Differential Equations," Appl. Numer. Math. **24** (2–3), 319–330 (1997).

19. D. Sarafyan, "Approximate Solution of Ordinary Differential Equations and Their Systems through Discrete and Continuous Embedded Runge–Kutta Formulae and Upgrading of Their Order," Comput. Math. Appl. **28** (10–12), 353–384 (1994).

20. C. A. H. Paul, *A User-Guide to Archi: An Explicit Runge–Kutta Code for Solving Delay and Neutral Differential Equations and Parameter Estimation Problems*, Numerical Analysis Report No. 283 (Extended) (Univ. of Manchester, Manchester, 1997).

21. C. T. H. Baker, J. C. Butcher, and C. A. H. Paul, *Experience of STRIDE Applied to Delay Differential Equations*, Numerical Analysis Report No. 208 (Univ. of Manchester, Manchester, 1992).

22. E. Lo and Z. Jackiewicz, "The Algorithm SNDDELM for the Numerical Solution of Systems of Neutral Delay Differential Equations," in *Delay Differential Equations with Applications in Population Dynamics* (Academic Press, New York, 1993), pp. 377–393.

23. L. F. Shampine and S. Thompson, "Solving DDEs in Matlab," Appl. Numer. Math. **37** (4), 441–458 (2001).

24. MathWorks. https://www.mathworks.com/. Cited February 9, 2020.

25. O. J. Dahl, E. W. Dijkstra, and C. A. R. Hoare, *Structured Programming* (Academic Press, New York, 1972).