УДК 519.6

РЕАЛИЗАЦИЯ ВЕКТОРИЗОВАННЫХ КОНЕЧНО-РАЗНОСТНЫХ АЛГОРИТМОВ РЕШЕНИЯ КРАЕВЫХ ЗАДАЧ МЕХАНИКИ ЖИДКОСТИ И ГАЗА В ПАКЕТЕ MATLAB

К. Н. Волков[1], В. Н. Емельянов[1]

Рассматривается подход к организации векторизованных вычислений и реализации конечно-разностных методов решения краевых задач механики жидкости и газа в пакете MATLAB. Приводятся конкретные примеры и обсуждаются особенности программной реализации разработанных вычислительных алгоритмов.

Ключевые слова: краевая задача, векторизованные алгоритмы, конечно-разностные схемы, программирование, пакеты вычислительной математики, динамика жидкости.

1. Введение. Многие задачи механики жидкости и газа при их математическом моделировании сводятся к краевым задачам для систем дифференциальных уравнений. В то время как численное решение задачи Коши хорошо формализуется и обеспечивается надежно разработанными методами нахождения приближенного решения [1, 2], а широкий набор этих методов (методы Рунге–Кутты, Адамса, Гира и др.) представлен во многих пакетах вычислительной математики, краевые задачи, в общем случае, представляют собой более сложный объект для исследования. Решение многих из них требует индивидуального подхода и выбора наиболее эффективного метода решения.

Систематическое использование возможностей, предоставляемых новыми информационными технологиями, средствами объектно-ориентированного программирования и современными операционными системами, требует рассмотрения особенностей решения краевых задач при помощи таких оболочек.

Одной из распространенных и популярных систем инженерных и математических расчетов является пакет MATLAB (MATrix LABoratory), разрабатываемый компанией Mathworks, Inc. (сайт компании — www.mathworks.com) и ставший фактически стандартом учебного программного обеспечения в области математического моделирования.

2. Преимущества пакета. Система MATLAB обладает рядом преимуществ перед другими программными средами, предназначенными для выполнения научных и инженерных расчетов. Среди таких преимуществ можно отметить следующие:

— многофункциональность, открытость, простая расширяемость и приспособляемость системы к решению необходимого класса задач;

— использование проблемно-ориентированных функций, предоставляющих широкие возможности для решения задач, характерных для конкретной научной отрасли;

— реализация современных возможностей объектно-ориентированного и визуального программирования;

— использование в качестве структур данных векторов и матриц (в пакете каждая заданная переменная представляется в виде матрицы 1×1), высокая скорость выполнения встроенных векторных и матричных операторов;

— наличие нового типа данных — разреженной матрицы и встроенных функций для работы с ними (операции хранения, преобразования, упорядочения, графического представления и решения систем линейных уравнений), высокая скорость выполнения вычислений с разреженными матрицами большой размерности;

— возможность создания векторизованных алгоритмов за счет замены циклических вычислений векторными и матричными операциями, а также применения встроенных функций и использования логических операторов вместо операторов цикла;

— выполнение вычислений как с плавающей точкой, так и применение символьной обработки данных (при помощи пакета Symbolic Mathematics Toolbox);

[1] Балтийский государственный технический университет “Военмех” им. Д. Ф. Устинова, физико-механический факультет, 1-я Красноармейская ул., д. 1, 190005, Санкт-Петербург; e-mail: kvolkov@kv7340.spb.edu

— наличие объектной графической системы (Handle Graphics) и графического интерфейса пользователя (Graphics User Interface);

— возможности высококачественной визуализации двух- и трехмерных графических изображений, мультипликации и звуковой интерпретации данных, импорт/экспорт графических, аудио- и видео-файлов в различных форматах;

— реализация удобной, универсальной и простой в применении интегрированной среды, которая позволяет формулировать задачи и получать их решение в привычной математической форме, не прибегая к рутинному программированию;

— подключение специализированных пакетов (Toolbox), расширяющих возможности системы (решение уравнений в частных производных, решение задач оптимизации, обработка сигналов, выполнение картографических работ, проведение финансовых вычислений и многие другие);

— наличие режима непосредственных вычислений и режима программирования на специальном языке;

— краткость и обозримость программы, перевод написанной программы в программу на языке C/C++, а также создание расширений пакета на языке C/C++;

— возможность сохранения и загрузки переменных в рабочую область пакета из специального файла;

— реализация пакета для различных операционных систем (Windows, VAX, UNIX);

— доступность системы (имеется студенческая версия пакета, в которую заложены некоторые ограничения по размеру используемых массивов, но доступная для массового пользователя по своей стоимости);

— сопровождение системы справочной документацией, представленной в pdf и html-формате.

Перечисленные достоинства превращают MATLAB в универсальное средство для проведения сложных научно-технических расчетов, а язык пакета — в мощный объектно-ориентированный язык программирования, близкий по своему синтаксису к языку программирования C/C++.

Среди недостатков пакета можно отметить выполнение программ в режиме интерпретации. Однако при грамотном программировании, учитывающем особенности среды MATLAB, интерпретационный характер организации вычислений практически не замечается.

3. Использование пакета для решения краевых задач. Основу программных средств, ориентированных на численное решение краевых задач, составляют конечно-разностные методы. Применение системы MATLAB открывает новые возможности по формализации и реализации конечно-разностных методов решения краевых задач для систем дифференциальных уравнений. Отметим, в частности, возможности построения векторизованных алгоритмов и использование прямых методов решения систем линейных уравнений.

Описание интегрированной среды, синтаксиса команд, функций и элементов программирования приводится в специальной литературе [3 – 5]. Начиная с версии 6.x, в состав MATLAB входят стандартные функции решения двухточечных краевых задач для систем обыкновенных дифференциальных уравнений. В состав MATLAB входит также пакет PDE Toolbox, который содержит средства для исследования и решения дифференциальных уравнений в частных производных второго порядка при помощи метода конечных элементов [6, 7]. К основным достоинствам применения пакета PDE Toolbox можно отнести полноценный графический интерфейс и интерактивный режим работы при составлении модели, генерации сетки, решении и его визуализации (приложение PDETool); возможность задания граничных условий Дирихле и Неймана, а также смешанных граничных условий; гибкость постановки задачи; полностью автоматическое разбиение расчетной области и адаптивный выбор величины конечных элементов; нелинейные и адаптивные расчетные схемы; возможность визуализации полей различных параметров. Встроенные функции MATLAB, команды и графический интерфейс пакета PDE Toolbox могут быть использованы для математического моделирования широкого класса инженерных и научных приложений.

При решении задач механики жидкости и газа конечно-элементный подход оказывается менее точным, чем метод конечных разностей или метод конечного объема. Тип дискретизации и используемый сеточный шаблон в значительной степени определяют свойства и особенности разностной схемы, а также вычислительной процедуры в целом (устойчивость, точность, время счета и другие показатели). Определенное влияние на показатели численного решения оказывают метод решения системы разностных уравнений, реализация граничных условий и ряд дополнительных факторов, характеризующих выбранную разностную схему (например, распределение узлов или методы ускорения сходимости). Во многих случаях требуется учитывать структуру исходной системы уравнений и сложную пространственно-временную зависимость правой части уравнения.

В данной работе рассматривается построение векторизованных алгоритмов решения краевых задач механики жидкости и газа, а также особенности их программной реализации в пакете MATLAB. Возможности разработанного подхода демонстрируются на примере решения автомодельной задачи динамики вязкой жидкости. Несмотря на это, предложенный подход применим и для решения более сложных задач, в частности, для интегрирования параболических дифференциальных уравнений в частных производных при помощи маршевой процедуры, а после некоторой модификации — для решения уравнений Навье–Стокса в переменных "функция тока–вихрь скорости". Реализованные алгоритмы, с одной стороны, широко используют функции MATLAB, предназначенные для обработки векторов и разреженных матриц, а с другой, отличаются высокой эффективностью и скоростью счета, сравнимыми с показателями программ, написанных на языке C/C++. Программы, реализованные в среде MATLAB, содержат всего лишь один итерационный цикл (невекторизуемая часть вычислительного алгоритма) и превосходят подобные невекторизованные (циклические) алгоритмы по скорости счета в несколько раз.

4. Адресация значений сеточной функции. Разностная стека, покрывающая расчетную область, имеет структуру, схожую со структурой двумерного массива, что позволяет производить адресацию доступа к ее ячейкам или узлам через систему двух индексов i и j. Узел сетки с индексами (i, j) имеет соседей, к которым легко адресоваться, давая соответствующие приращения индексов. Указанный принцип адресации работает как для обозначения узлов сетки в методе конечных разностей, так и для обозначения ячеек в методе контрольного объема.

4.1. Векторизация циклов. При программировании вычислительных задач работа с сеточными структурами обычно осуществляется при помощи переборного алгоритма, реализованного на основе вложенных циклов. На условном языке программирования алгоритм перебора элементов массива можно записать в следующем виде:

<table>
<tr><th>одномерный случай</th><th>двумерный случай</th></tr>
<tr><td><pre>for i=1 to nx step 1
  begin
    U(i)=...
  end</pre></td><td><pre>for i=1 to nx step 1
  for j=1 to ny step 1
    begin
      U(i,j)=...
    end</pre></td></tr>
</table>

Данный подход подразумевает вычисление индексного выражения для адресации к данным и занесение результатов вычислений в массив `U`.

В задачах механики жидкости и газа двумерным структурам данных соответствуют поля гидродинамических величин и координаты сеточных узлов при решении задачи в области прямоугольной формы или в криволинейной области, которую можно отобразить на вычислительный прямоугольник при помощи соответствующего преобразования координат [8].

Многомерные массивы в оперативной памяти компьютера хранятся в виде линейных последовательных структур, а аппаратные возможности современных процессоров позволяют обеспечить высокую производительность поточных вычислений конвейерного типа при соответствующей организации данных. Принципы кэширования и буферизации, заложенные в архитектуру современных процессорных устройств, позволяют осуществить векторизацию вычислительных алгоритмов на "обычных", а не векторных компьютерах.

Векторизация позволяет работать со множеством данных как с единой вычислительной структурой, что не только делает компактным запись вычислительного алгоритма, избегая вложенных циклов, но и повышает эффективность вычислений. Организовать векторизацию вычислений можно при помощи средств современных объектно-ориентированных языков программирования, в частности, при помощи пакета MATLAB.

4.2. Адресация к внутренним ячейкам. При разностной дискретизации дифференциальных операторов, фигурирующих в уравнениях и граничных условиях, кроме текущего избираемого данного необходимо обеспечить адресацию к данным в соседних узлах, входящих в вычислительный шаблон. Традиционный способ адресации к таким данным состоит в указании двух индексов, определяющих столбец и строку выбираемого данного.

Пронумеруем ячейки расчетной сетки сквозным образом (например, по столбцам). Тогда при помощи одномерного массива индексов возможно указать определенное подмножество ячеек, покрывающих расчетную область (рис. 1). Например, обозначив через **pC** вектор (множество) индексов всех внутренних ячеек (фрагмент рисунка а), можно указать подмножества ячеек, которые лежат выше (фрагмент б) или ниже заданного слоя ячеек, прибавив или отняв единицу от всех компонент вектора **pC** и образовав для обозначения этих ячеек вектора

$$\mathbf{pU} = \mathbf{pC} + 1; \quad \mathbf{pD} = \mathbf{pC} - 1.$$

Таким же способом можно определить индексные вектора для всех левых (фрагмент в) и правых ячеек расчетной области:

$$\mathbf{pL} = \mathbf{pC} - m; \quad \mathbf{pR} = \mathbf{pC} + m.$$

Величина m определяет количество ячеек (число элементов индексного вектора) в столбце расчетной области.

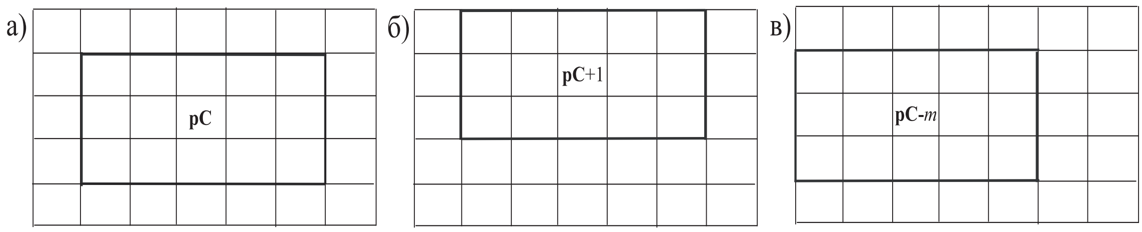
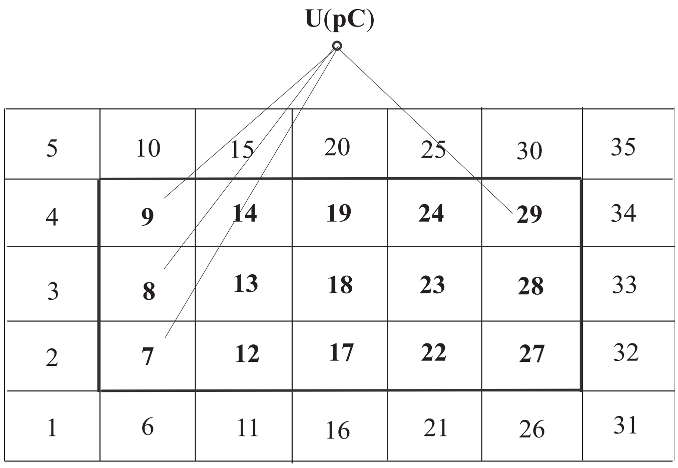


Рис. 1. Адресация к подмножествам ячеек в расчетной области прямоугольной формы

4.3. Адресация к граничным ячейкам. Для постановки граничных условий достаточно часто выделяются фиктивные ячейки, образующие вместе с внутренними ячейками расширенную расчетную область. Внутренние ячейки выделены полужирным шрифтом на рис. 2, а адресация к ним обеспечивается с помощью индексного вектора **pC**.



pC={7, 8, 9, 12, 13, 14, 17, 18, 19, 22, 23, 24, 27, 28, 29}'

Рис. 2. Расширенная область ячеек

Способ адресации к фиктивным граничным ячейкам поясняет рис. 3. Для этого строятся индексные вектора **pBL**, **pBU**, **pBR**, **pBD**, соответствующие левой, верхней, правой и нижней границам расчетной области. Способ их формирования такой же, как и индексных векторов внутренних ячеек (угловые ячейки при этом не используются).

4.4. Вычисление потоков. В качестве объекта, которому ставится в соответствие поток, выбирается грань, разделяющая две соседние ячейки. Привязка потока к грани позволяет избежать его повторного вычисления при реализации законов сохранения для контрольного объема. Одна и та же величина потока участвует в построении балансовых соотношений для двух соседних ячеек (входящий в ячейку поток является выходящим для соседней ячейки), что обеспечивает консервативность разностной схемы. При введении фиктивных ячеек все ячейки, расположенные внутри области, имеют соседей, и вычисление потоков для всех граней проводится по единому алгоритму.

Для вычисления потоков формируется два множества индексных векторов (рис. 4). Один из таких векторов **pCFL** определяет множество всех ячеек, расположенных слева от боковых граней (фрагмент а).

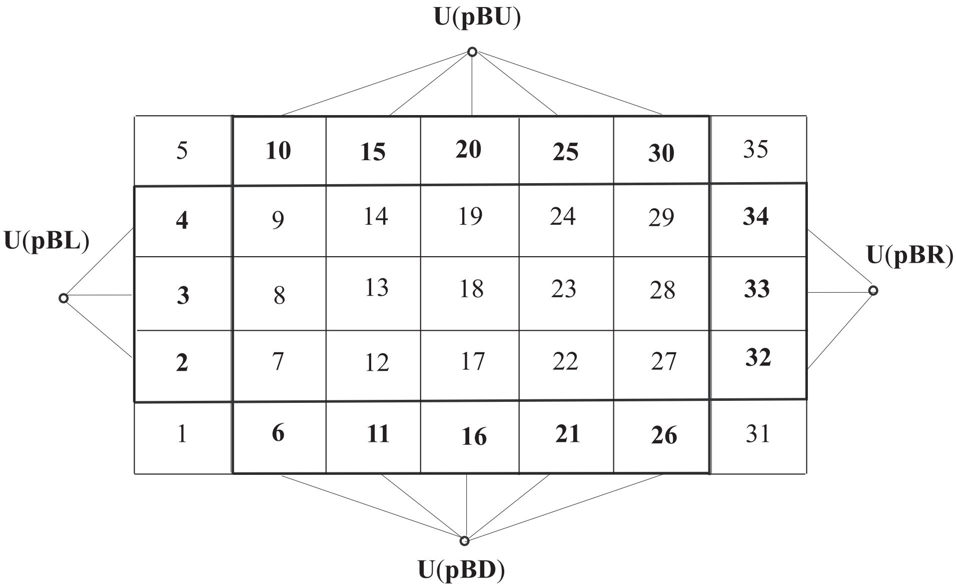
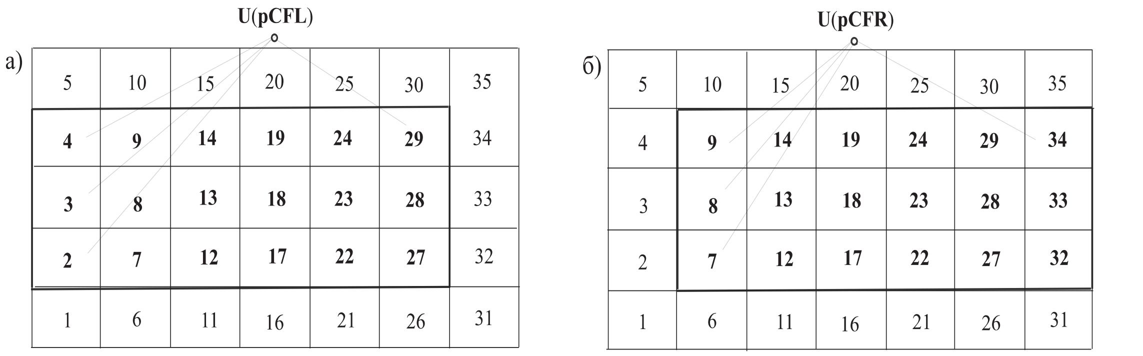


Рис. 3. Индексные вектора, соответствующие фиктивным ячейкам



pCFL={2, 3, 4, 7, 8, 9, 12, 13, 14, 17, 18, 19, 22, 23, 24, 27, 28, 29}' **pCFR**={7, 8, 9, 12, 13, 14, 17, 18, 19, 22, 23, 24, 27, 28, 29, 32, 33, 34}'

Рис. 4. Множества левых и правых ячеек расчетной области при вычислении потоков

Другой индексный вектор **pCFR** обеспечивает операцию подстановочной адресации для множества всех правых ячеек (фрагмент б) и определяется следующим образом:

$$\mathbf{pCFR} = \mathbf{pCFL} + m.$$

Векторная операция вычисления потоков задается соотношением

$$\mathbf{F}(\mathbf{pC}) = \mathbf{F}\big(\mathbf{U}(\mathbf{pCFL}),\, \mathbf{U}(\mathbf{pCFR})\big). \tag{1}$$

Последовательность вычислений, записанная в обычных индексных выражениях, имеет вид

$$\mathbf{F}_{i+1/2,j} = \mathbf{F}(\mathbf{U}_{i+1,j},\, \mathbf{U}_{i,j}). \tag{2}$$

Индексы i и j относятся к центру ячейки, а индекс $i+1/2$ соответствует грани, разделяющей ячейки с индексами i и $i+1$.

Отметим принципиальное различие между выражениями (1) и (2). В то время как соотношение (1) определяет все множество потоков, соотношение (2) требует двойного цикла вычислений по индексам i и j. Преимущества подхода, основанного на выражении (1), проявляются при применении вычислительных сред, допускающих выполнение векторизованных операций (в частности, пакета MATLAB или средств объектно-ориентированных языков программирования, например C++).

Поскольку поток привязывается к грани ячейки, а параметры газа — к ее центру (рис. 5), то необходимо определить соответствие индексных выражений для вычисления потоков с индексами, определяющими ячейки расчетной области. Несмотря на то, что поток содержит меньшее количество элементов по сравнению с расширенным множеством ячеек, имеет смысл хранить его в структуре той же размерности (хотя ряд элементов при этом и не используется, но удобства адресации и наглядность превалируют над соображениями экономии памяти).

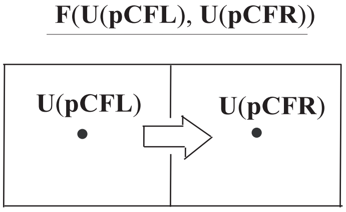


Рис. 5. Векторизованное вычисление потока

5	10	15	20	25	30	35
4	**9**	**14**	**19**	**24**	**29**	34
3	**8**	**13**	**18**	**23**	**28**	33
2	**7**	**12**	**17**	**22**	**27**	32
1	6	11	16	21	26	31

pC={2, 3, 4, 7, 8, 9, 12, 13, 14, 17, 18, 19, 22, 23, 24, 27, 28, 29}'

Рис. 6. Размещение потоков и структура вектора ссылок

Сформируем индексный вектор для определения множества потоков через боковые грани расчетных ячеек исходя из схемы размещения потоков, приведенной на рис. 6.

При такой организации данных легко записываются векторизованные балансовые соотношения для каждой ячейки расчетной области. Адресация к потокам через боковые грани множества ячеек показана на рис. 7. Для всех расчетных ячеек, определяемых индексным вектором **pC**, потоки через их левые грани определяются индексным вектором $\mathbf{pC} - m$, а через правые грани — индексным вектором **pC**. Величина m представляет собой количество ячеек в столбце расширенной области ячеек и определяет сдвиг при переходе от двумерной матричной структуры к линейному размещению элементов.

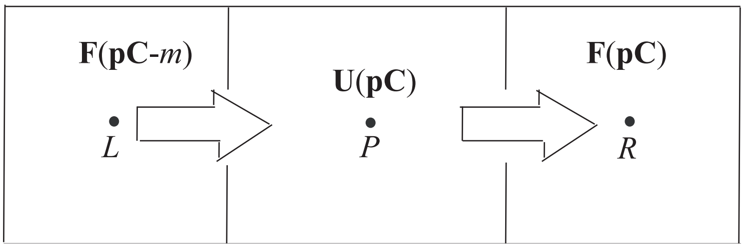


Рис. 7. Вычисление потоков через боковые грани ячейки

5. Вычисление производных. Рассмотрим конечно-разностную дискретизацию производной от сеточной функции **U**, определенной на последовательном наборе из N равноотстоящих точек. В программе такой набор данных представляется при помощи линейного массива, содержащего N элементов.

Для вычисления производной в MATLAB используется функция `y=diff(x,n)`, которая вычисляет конечные разности порядка n, удовлетворяющие рекуррентному соотношению `diff(x,n)=diff(x,n-1)`. Если `x` — одномерный массив вида `x=[x(1) x(2) ... x(n)]`, то `diff(x)` представляет собой вектор разностей соседних элементов `diff(x)=[x(2)-x(1) x(3)-x(2) ... x(n)-x(n-1)]`. Количество элементов вектора `x` на единицу меньше количества элементов вектора `diff(x)`. Приближением производной порядка n является `diff(y,n)/diff(x,n)`.

Вычислить производную можно и другим способом, используя описанный способ адресации к сеточным структурам данных. Разобьем промежуток интегрирования $x_0 \leqslant x \leqslant x_N$ на N равных отрезков, а длину каждого из них обозначим через $\Delta x = x_i - x_{i-1}$ (рис. 8). Пронумеруем все промежутки разностного шаблона от 1 до $N-1$. Введем массив индексов **I**, в котором последовательное возрастание индексов соответствует порядку следования его элементов:

$$\mathbf{I} = \{1, 2, \ldots, P-1, P, P+1, \ldots, N-1, N\}' = 1:N.$$

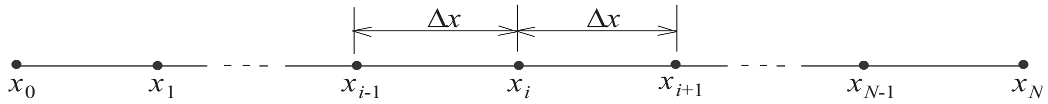


Рис. 8. Разбиение промежутка интегрирования по координате x

Выберем разностный шаблон так, как показано на рис. 9. Срединную точку каждого отрезка обозначим через P, а левую и правую границы интервала — через L и R соответственно (всего имеется $N-1$ точек P, L и R). Дополнительно введем массивы индексов для левых и правых элементов сеточной структуры:

$$\mathbf{L} = \{1, 2, \ldots, P-1, P, P+1, \ldots, N-2, N-1\}' = 1 : N-1;$$
$$\mathbf{R} = \{2, 3, \ldots, P-1, P, P+1, \ldots, N-1, N\}' = 2 : N.$$

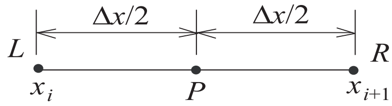


Рис. 9. Разностный шаблон

При условии, что соответствующим программно-математическим обеспечением поддерживается векторная операция вычитания

$$\Delta\mathbf{U} = \mathbf{U}(\mathbf{R}) - \mathbf{U}(\mathbf{L}),$$

понимаемая как

$$\begin{pmatrix} \Delta\mathbf{U}(1) \\ \Delta\mathbf{U}(2) \\ \vdots \\ \Delta\mathbf{U}(N-1) \end{pmatrix} = \begin{pmatrix} \mathbf{U}\big(\mathbf{R}(1)\big) - \mathbf{U}\big(\mathbf{L}(1)\big) \\ \mathbf{U}\big(\mathbf{R}(2)\big) - \mathbf{U}\big(\mathbf{L}(2)\big) \\ \vdots \\ \mathbf{U}\big(\mathbf{R}(N-1)\big) - \mathbf{U}\big(\mathbf{L}(N-1)\big) \end{pmatrix},$$

формула векторизованного вычисления производной имеет вид

$$\frac{d\mathbf{U}(x)}{dx} = \frac{\mathbf{U}(\mathbf{R}) - \mathbf{U}(\mathbf{L})}{\Delta x} + O(\Delta x^2).$$

В MATLAB алгоритм вычисления производной можно представить в виде следующей последовательности шагов:

```
% индексы промежутков и их граничных узлов
iI=1:N-1; iL=iI; iR=iI+1;
% длина промежутка
dx=1/(N-1);
% производная y'(x)
yp=(y(iR)-y(iL))/dx;
```

Приведенный способ адресации к сеточным структурам данных лежит в основе вычисления производных от сеточных функций и построения векторизованных алгоритмов решения краевых задач.

6. Формулировка краевой задачи. Рассмотрим систему обыкновенных дифференциальных уравнений порядка M, разрешенную относительно производных:

$$\frac{d\mathbf{U}}{dx} + F\,\mathbf{U} = \mathbf{H}, \tag{3}$$

где $\mathbf{U}$ — вектор-столбец неизвестных, F — матрица коэффициентов, $\mathbf{H}$ — вектор-столбец правых частей. Для корректного решения системы (3) необходимо, чтобы число краевых условий равнялось порядку системы. В общем случае краевые условия между начальной x_0 и конечной x_N точками промежутка интегрирования распределяются в произвольном соотношении.

Каждому узлу расчетной области x_i $(i = 0, \ldots, N)$ поставим в соответствие вектор физических переменных $\mathbf{U}_i$. Вектора $\mathbf{U}_0$ и $\mathbf{U}_N$, соответствующие началу и концу промежутка интегрирования, содержат в себе левые и правые граничные значения.

Для каждой точки P запишем уравнение (3) в дискретном виде:

$$\frac{\mathbf{U}_R - \mathbf{U}_L}{\Delta x} + F_P \mathbf{U}_P = \mathbf{H}_P. \tag{4}$$

Матрицу коэффициентов F_P и вектор правых частей $\mathbf{H}_P$, соответствующие узлу P, вычислим как полусумму величин, относящихся к точкам L и R, а именно

$$F_P \mathbf{U}_P = \frac{1}{2}\left(F_L \mathbf{U}_L + F_R \mathbf{U}_R\right), \quad \mathbf{H}_P = \frac{1}{2}\left(\mathbf{H}_L + \mathbf{H}_R\right).$$

После подстановки в (4) получим

$$\mathbf{U}_R - \mathbf{U}_L + \frac{\Delta x}{2}\left(F_L \mathbf{U}_L + F_R \mathbf{U}_R\right) = \frac{\Delta x}{2}\left(\mathbf{H}_L + \mathbf{H}_R\right). \tag{5}$$

Для линеаризации (5) используется метод замороженных коэффициентов.

7. Граничные условия. Для построения удобных в работе программных средств важно удачно формализовать постановку граничных условий.

Введем для левой и правой границы индексные вектора $\mathbf{I}_L$ и $\mathbf{I}_R$, которые определяются следующим образом:

$$\mathbf{I}_L = \begin{pmatrix} I_1 \\ I_2 \\ \vdots \\ I_M \end{pmatrix}, \quad I_k = \begin{cases} 1, & \text{если задано левое граничное условие для } k\text{-й компоненты вектора решения;} \\ 0, & \text{если левое граничное условие для } k\text{-й компоненты вектора решения отсутствует;} \end{cases}$$

$$\mathbf{I}_R = \begin{pmatrix} I_1 \\ I_2 \\ \vdots \\ I_M \end{pmatrix}, \quad I_k = \begin{cases} 1, & \text{если задано правое граничное условие для } k\text{-й компоненты вектора решения;} \\ 0, & \text{если правое граничное условие для } k\text{-й компоненты вектора решения отсутствует.} \end{cases}$$

Сформируем два вспомогательных вектора $\mathbf{L}$ и $\mathbf{R}$, компоненты которых содержат на местах, определяемых индексными векторами $\mathbf{I}_L$ и $\mathbf{I}_R$, левые и правые граничные значения соответственно, а остальные их компоненты имеют неопределенные (произвольные) значения. Произвольно заданные компоненты векторов $\mathbf{L}$ и $\mathbf{R}$ не принимают участия в вычислениях. Введем также две целочисленные характеристики M_L и M_R, равные количеству левых и правых граничных условий (очевидно, что $M_L + M_R = M$).

8. Вектора вычислительных переменных. Вектора $\mathbf{U}_0$ и $\mathbf{U}_N$ на некоторых, в общем случае произвольных местах содержат величины, соответствующие граничным условиям, что затрудняет формализацию расчетной процедуры с векторами физических переменных.

Введем вектора вычислительных переменных $\mathbf{W}_i$ (их количество на единицу меньше, чем физических векторов $\mathbf{U}_i$), которые содержат только неизвестные. Каждый вычислительный вектор $\mathbf{W}_i$ содержит часть неизвестных из вектора $\mathbf{U}_{i-1}$ и часть неизвестных из вектора $\mathbf{U}_i$. Очевидно, что любая из компонент векторов $\mathbf{U}_i$ входит в набор векторов $\mathbf{W}_i$ не более одного раза. При этом известные компоненты векторов $\mathbf{U}_0$ и $\mathbf{U}_N$, соответствующие граничным условиям, в вектора $\mathbf{W}_i$ не входят.

Рассмотрим общую схему построения вычислительных векторов $\mathbf{W}_i$. Построение начнем с вектора $\mathbf{W}_N$. Компоненты вектора $\mathbf{U}_N$, не соответствующие граничным условиям, переносятся в вектор $\mathbf{W}_N$ на те же места. Оставшиеся в векторе $\mathbf{W}_N$ компоненты заполняются компонентами вектора $\mathbf{U}_{N-1}$ с тех же позиций. Формально данную процедуру можно записать с помощью вектора индексов $\mathbf{I}_R$, а именно

$$\mathbf{W}_N^j = \left(1 - \mathbf{I}_R^j\right) \mathbf{U}_N^j + \mathbf{I}_R^j \mathbf{U}_{N-1}^j. \tag{6}$$

Двигаясь влево от правой границы (индексы перебираются от N до 1), вектор $\mathbf{W}_i$ представим в виде

$$\mathbf{W}_i^j = \left(1 - \mathbf{I}_R^j\right) \mathbf{U}_i^j + \mathbf{I}_R^j \mathbf{U}_{i-1}^j. \tag{7}$$

Метод формирования вектора $\mathbf{W}_1$ отличается от остальных. Невостребованные еще компоненты вектора $\mathbf{U}_1$ займут свои позиции, а остальные места необходимо заполнить компонентами вектора $\mathbf{U}_0$, не соответствующими граничным условиям. Для этого нужно взять эти компоненты в порядке возрастания их индексов и поместить на свободные места вектора $\mathbf{W}_1$ в порядке возрастания индексов этих мест. Представление вектора $\mathbf{W}_1$ имеет вид

$$\mathbf{W}_1^j = \left(1 - \mathbf{I}_R^j\right) \mathbf{U}_1^j + \mathbf{I}_R^j \mathbf{U}_*^j. \tag{8}$$

Вектор $\mathbf{U}_*$ получается из вектора $\mathbf{U}_0$ при помощи перестановки компонент согласно описанному принципу. Формальное представление вектора $\mathbf{U}_*$ непринципиально для дальнейшего изложения и здесь не приводится.

9. Формулы перехода. Выразим вектор физических переменных через вектора вычислительных переменных. Учитывая соглашение о том, что вычислительный вектор формируется с учетом формы задания правых граничных условий, для $i = 2, \ldots, N-1$ получим

$$\mathbf{U}_i = T_L^i \mathbf{W}_i + T_R^i \mathbf{W}_{i+1}. \tag{9}$$

Соотношение (9) формализует простой сдвиг компонентов, который делается таким образом, чтобы компоненты вычислительных векторов попали на свои места в физических векторах.

Правая матрица перестановок T_R содержит компоненты индексного вектора $\mathbf{I}_R$ на главной диагонали, а остальные компоненты этой матрицы равны нулю:

$$T_R^i = \mathbf{I}_R E = \begin{pmatrix} I_R^1 & 0 & \cdots & 0 \\ 0 & I_R^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & I_R^M \end{pmatrix}.$$

Левая матрица перестановок T_L содержит нули на главной диагонали там, где соответствующие компоненты индексного вектора $\mathbf{I}_R$ равны единице, и единицы в противном случае, а остальные компоненты этой матрицы равны нулю:

$$T_L^i = E - T_R^i = \begin{pmatrix} 1 - I_R^1 & 0 & \cdots & 0 \\ 0 & 1 - I_R^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 - I_R^M \end{pmatrix}.$$

Матрицы перестановок T_L^i и T_R^i одинаковы для всех внутренних точек области интегрирования (при $i = 1, \ldots, N-1$).

Для правой границы (при $i = N$) представление вектора физических переменных имеет вид

$$\mathbf{U}_N = T_L^N \mathbf{W}_N + T_R^N \mathbf{R}. \tag{10}$$

Матрицы перестановок T_L^N и T_R^N остаются такими же, что и во внутренних точках.

Особый вид имеет представление вектора физических переменных на левой границе (при $i = 0$). Для формирования соответствующего выражения введем несколько вспомогательных матричных объектов. Заметим, что квадратная матрица, имеющая единственный ненулевой элемент, равный 1 на l-й строке в k-м столбце, при умножении на вектор-столбец переместит k-й элемент вектора в l-е положение. Представление вектора физических переменных для точки $i = 0$ связано с видом вычислительного вектора при $i = 1$. Структура этого соотношения имеет форму

$$\mathbf{U}_0 = T_L^0 \mathbf{L} + T_R^0 \mathbf{W}_1. \tag{11}$$

Первый член в правой части представляет собой набор компонент вектора переменных, которые определяются граничными значениями на левом конце промежутка. Второй член представляет те компоненты вектора, которые определяются в процессе решения. Левая матрица перестановок имеет вид

$$T_L^0 = E - \mathbf{I}_L E.$$

Матрица T_R^0 обеспечивает подстановку тех переменных, которые не входят в число граничных условий и поэтому находятся в векторе $\mathbf{W}_1$.

Для формирования матрицы T_R^0 вектор $\mathbf{W}_1$ заполняется теми компонентами вектора $\mathbf{U}_1$, для которых не заданы правые граничные условия (эти компоненты формируются так же, как в прочих векторах). В результате оказываются определенными $M_L = M - M_R$ компонент. Для формирования остальных M_R компонент находится первый нулевой элемент I_L^j вектора $\mathbf{I}_L$ и первый отличный от нуля элемент I_R^k вектора $\mathbf{I}_R$. Пара индексов (j,k) определяет закон переноса элемента из $\mathbf{U}_0$ в $\mathbf{W}_1$. Далее определяется следующая пара индексов и т.д. Построенное множество из M_R пар индексов позволяет сформировать матрицу T_R^0, которая содержит единицы на местах, указываемые парами индексов (первое число пары указывает строку, а второе число — столбец); остальные ее элементы равны нулю. Описанная выше процедура позволяет связать матричным соотношением физический и вычислительный вектора в точке $i = 0$.

10. Разностная схема в вычислительных переменных. В вычислительных переменных разностная схема для уравнения (3) примет вид

$$\mathbf{U}_i - \mathbf{U}_{i-1} + \frac{\Delta x}{2}\left(F_{i-1}\mathbf{U}_{i-1} + F_i\mathbf{U}_i\right) = \frac{\Delta x}{2}\left(\mathbf{H}_{i-1} + \mathbf{H}_i\right). \tag{12}$$

Учитывая представления вектора физических переменных через вектор вычислительных переменных, после подстановки в (12) имеем

$$\begin{aligned} &\Big(-T_L^{i-1} + \frac{\Delta x}{2}F_{i-1}T_L^{i-1}\Big)\mathbf{W}_{i-1} + \Big[T_L^i - T_R^{i-1} + \frac{\Delta x}{2}\left(F_{i-1}T_R^{i-1} + F_iT_L^i\right)\Big]\mathbf{W}_i + \\ &\qquad + \Big(T_i^R + \frac{\Delta x}{2}F_iT_R^i\Big)\mathbf{W}_{i+1} = \frac{\Delta x}{2}(\mathbf{H}_{i-1} + \mathbf{H}_i). \end{aligned} \tag{13}$$

В матричном виде система разностных уравнений имеет вид

$$A\,\mathbf{U} = \mathbf{B}. \tag{14}$$

Здесь A — квадратная матрица коэффициентов размером $MN \times MN$, $\mathbf{U}$ и $\mathbf{B}$ — вектор-столбец неизвестных и вектор-столбец правых частей размером $M \times N$ соответственно. Для решения системы уравнений (14) можно воспользоваться одним из методов решения систем линейных уравнений, в частности методом блочной прогонки.

11. Метод прогонки. Соотношение (13) можно записать в виде

$$A_i\mathbf{W}_{i-1} + B_i\mathbf{W}_i + C_i\mathbf{W}_{i+1} = \mathbf{D}_i. \tag{15}$$

Здесь

$$\begin{aligned} A_i &= -T_L^{i-1} + \frac{\Delta x}{2}A_{i-1}T_L^{i-1}; \quad & B_i &= T_L^i - T_R^{i-1} + \frac{\Delta x}{2}(A_{i-1}T_R^{i-1} + A_iT_L^i); \\ C_i &= T_i^R + \frac{\Delta x}{2}A_iT_R^i; & \mathbf{D}_i &= \frac{\Delta x}{2}(\mathbf{B}_{i-1} + \mathbf{B}_i). \end{aligned}$$

В качестве векторов $\mathbf{W}_0$ и $\mathbf{W}_{N+1}$ следует брать вектора $\mathbf{L}$ и $\mathbf{R}$ соответственно. При $i = 1$ и $i = N$ имеем

$$B_1\mathbf{W}_1 + C_1\mathbf{W}_2 = \mathbf{D}_1 - A_1\mathbf{L}; \quad A_N\mathbf{W}_{N-1} + B_N\mathbf{W}_N = \mathbf{D}_N - C_N\mathbf{R}.$$

Прогоночные соотношения строятся по следующей зависимости:

$$\mathbf{W}_{i-1} = K_{i-1}\mathbf{W}_i + \mathbf{P}_{i-1}.$$

После подстановки получим

$$\mathbf{W}_i = -\left(A_iK_{i-1} + B_i\right)^{-1}C_i\mathbf{W}_{i+1} + \left(A_iK_{i-1} + B_i\right)^{-1}\left(\mathbf{D}_i - A_i\mathbf{P}_{i-1}\right).$$

В результате представления для K_i и $\mathbf{P}_i$ при $i > 1$ имеют вид

$$K_i = -\left(A_i K_{i-1} + B_i\right)^{-1} C_i; \quad \mathbf{P}_i = \left(A_i K_{i-1} + B_i\right)^{-1} \left(\mathbf{D}_i - A_i \mathbf{P}_{i-1}\right).$$

Матрица K_1 и вектор $\mathbf{P}_1$ вычисляются по соотношениям

$$K_1 = -B_i^{-1} C_i; \quad \mathbf{P}_1 = B_i^{-1} \left(\mathbf{D}_i - A_i \mathbf{L}\right).$$

Вектор $\mathbf{W}_N$ находится по формуле (прямой ход)

$$\mathbf{W}_N = \left(A_N K_{N-1} + B_N\right)^{-1} \left(\mathbf{D}_N - C_N \mathbf{R} - A_N \mathbf{P}_{N-1}\right).$$

Используя найденные значения прогоночных коэффициентов, вычисляются следующие значения векторов, двигаясь по расчетной области слева направо (обратный ход):

$$\mathbf{W}_i = K_i \mathbf{W}_{i+1} + \mathbf{P}_i.$$

Для заполнения физических векторов используются формулы перехода (9)–(11).

12. Пример краевой задачи. Рассмотрим течение вязкой несжимаемой жидкости вблизи неподвижной плоской стенки в случае, когда на большом расстоянии от стенки происходит вращение жидкости с постоянной угловой скоростью [9]. Расчет характеристик потока сводится к интегрированию следующей системы уравнений:

$$\begin{aligned}
&2F + H' = 0;\\
&F^2 - G^2 + HF' - F'' + 1 = 0;\\
&2GF + HG' - G'' = 0.
\end{aligned} \tag{16}$$

Граничные условия для системы (16) имеют вид

$$\begin{array}{llll}
F = 0, & G = 0, & H = 0 & \text{при} \quad x = 0;\\
F = 0, & G = 1 & & \text{при} \quad x = \infty.
\end{array}$$

Введем новые переменные

$$u_1 = F, \quad u_2 = F', \quad u_3 = G, \quad u_4 = G', \quad u_5 = H.$$

Тогда система уравнений (16) примет вид

$$\begin{aligned}
&\frac{du_1}{dx} = u_2;\\
&\frac{du_2}{dx} = u_1^2 - u_2 u_5 - u_3^2 + 1;\\
&\frac{du_3}{dx} = u_4;\\
&\frac{du_4}{dx} = 2u_1 u_3 + u_4 u_5;\\
&\frac{du_5}{dx} = -2u_1.
\end{aligned} \tag{17}$$

Граничные условия в новых переменных приобретают форму

$$\begin{array}{llll}
u_1 = 0, & u_3 = 0, & u_5 = 0 & \text{при} \quad x = 0;\\
u_1 = 0, & u_3 = 1 & & \text{при} \quad x = \infty.
\end{array} \tag{18}$$

Бесконечная область интегрирования по координате $x \in [0, \infty]$ заменяется отрезком конечной длины, верхняя граница которого выбирается исходя из условия гладкого сопряжения решения с граничным условием на бесконечности.

Линеаризуем систему уравнений (17). Разложим произведения неизвестных функций на итерации $n+1$ в ряд Тейлора в окрестности n-й итерации. С точностью до членов второго порядка малости получим

$$(u_1u_3)^{n+1} = (u_1u_3)^n + \left(\frac{\partial u_1u_3}{\partial x}\right)^n \Delta x = (u_1u_3)^n + \left(\frac{\partial u_1}{\partial x}u_3 + \frac{\partial u_3}{\partial x}u_1\right)^n \Delta x.$$

Производные от неизвестных функций по координате x на n-й итерации вычисляются по формулам

$$\left(\frac{\partial u_1}{\partial x}\right)^n = \frac{u_1^{n+1} - u_1^n}{\Delta x}, \quad \left(\frac{\partial u_3}{\partial x}\right)^n = \frac{u_3^{n+1} - u_3^n}{\Delta x}.$$

В результате на итерации $n+1$ получим

$$(u_1u_3)^{n+1} = u_1^{n+1}u_3^n + u_1^n u_3^{n+1} - u_1^n u_3^n.$$

Аналогичным образом можно показать, что

$$\begin{aligned} (u_1u_1)^{n+1} &= 2u_1^{n+1}u_1^n - u_1^n u_1^n; \quad (u_2u_5)^{n+1} = u_2^{n+1}u_5^n + u_2^n u_5^{n+1} - u_2^n u_5^n;\\ (u_3u_3)^{n+1} &= 2u_3^{n+1}u_3^n - u_3^n u_3^n; \quad (u_4u_5)^{n+1} = u_4^{n+1}u_5^n + u_4^n u_5^{n+1} - u_4^n u_5^n. \end{aligned}$$

Вектор-столбец неизвестных $\mathbf{U}$, матрица коэффициентов F и вектор-столбец правых частей системы $\mathbf{H}$ имеют вид

$$\mathbf{U} = \begin{pmatrix} u_1\\ u_2\\ u_3\\ u_4\\ u_5 \end{pmatrix}, \quad F = \begin{pmatrix} 0 & 1 & 0 & 0 & 0\\ 2u_1^n & u_5^n & -2u_3^n & 0 & u_2^n\\ 0 & 0 & 0 & 1 & 0\\ 2u_3^n & 0 & 2u_1^n & u_5^n & u_4^n\\ -2 & 0 & 0 & 0 & 0 \end{pmatrix}, \quad \mathbf{H} = \begin{pmatrix} 0\\ -u_1^n u_1^n - u_2^n u_5^n + u_3^n u_3^n + 1\\ 0\\ -2u_1^n u_3^n - u_4^n u_5^n\\ 0 \end{pmatrix}.$$

Для наглядности запишем систему уравнений (5) в скалярной форме:

$$\begin{aligned}
& u_{1R} - u_{1L} - \frac{\Delta x}{2}\,(u_{2R} + u_{2L}) = 0;\\
& u_{2R}\Big[1 - \frac{\Delta x}{4}\,(u^o_{5R} + u^o_{5L})\Big] - u_{2L}\Big[1 + \frac{\Delta x}{4}\,(u^o_{5R} + u^o_{5L})\Big] - \frac{\Delta x}{2}\Big\{u_{5R}\,(u^o_{2R} + u^o_{2L}) +\\
&\quad + u_{5L}\,(u^o_{2R} + u^o_{2L}) - 2\big[u_{3R}\,(u^o_{3R} + u^o_{3L}) + u_{3L}\,(u^o_{3R} + u^o_{3L})\big] + 2\big[u_{1R}\,(u^o_{1R} + u^o_{1L}) +\\
&\quad + u_{1L}\,(u^o_{1R} + u^o_{1L})\big]\Big\} = \Delta x - \frac{\Delta x}{4}\Big[(u^o_{2R} + u^o_{2L})\,(u^o_{5R} + u^o_{5L}) - (u^o_{3R} + u^o_{3L})^2 + (u^o_{1R} + u^o_{1L})^2\Big];\\
& u_{3R} - u_{3L} - \frac{\Delta x}{2}\,(u_{4R} + u_{4L}) = 0;\\
& u_{4R}\Big[1 - \frac{\Delta x}{4}\,(u^o_{5R} + u^o_{5L})\Big] - u_{4L}\Big[1 + \frac{\Delta x}{4}\,(u^o_{5R} + u^o_{5L})\Big] - \frac{\Delta x}{2}\Big\{2\big[u_{3R}\,(u^o_{1R} + u^o_{1L}) +\\
&\quad + u_{3L}\,(u^o_{1R} + u^o_{1L}) + u_{1R}\,(u^o_{3R} + u^o_{3L}) + u_{1L}\,(u^o_{3R} + u^o_{3L})\big] + u_{5R}\,(u^o_{4R} + u^o_{4L}) +\\
&\quad + u_{5L}\,(u^o_{4R} + u^o_{4L})\Big\} = \Delta x\Big[(u^o_{4R} + u^o_{4L})\,(u^o_{5R} + u^o_{5L}) + 2\,(u^o_{3R} + u^o_{3L})\,(u^o_{1R} + u^o_{1L})\Big];\\
& u_{5R} - u_{5L} - \Delta x\,(u_{1R} + u_{1L}) = 0.
\end{aligned}$$

Верхний индекс o соответствует решению, полученному на предыдущей итерации.

В каждой внутренней точке P имеется по 5 неизвестных. Вектора $\mathbf{U}$ и $\mathbf{B}$ имеют следующий вид:

$$\mathbf{U} = \Big\{u_1^{(1)}, u_3^{(1)}, u_5^{(1)}, u_1^{(2)}, u_2^{(2)}, \ldots, u_5^{(2)}, \ldots, u_1^{(N-1)}, u_2^{(N-1)}, \ldots, u_5^{(N-1)}, u_1^{(N)}, u_3^{(N)}\Big\}';$$

$$\mathbf{B} = \Big\{b_1^{(1)}, b_3^{(1)}, b_5^{(1)}, b_1^{(2)}, b_2^{(2)}, \ldots, b_5^{(2)}, \ldots, b_1^{(N-1)}, b_2^{(N-1)}, \ldots, b_5^{(N-1)}, b_1^{(N)}, b_3^{(N)}\Big\}'.$$

Матрица A является разреженной, ее ненулевые элементы группируются около главной диагонали. Первые три и последние две строки матрицы A представляют собой левые и правые граничные условия.

Остальные строки выражают собой исходную систему уравнений во внутренних точках вычислительной области. Такой же порядок элементов имеет место в векторе неизвестных **U** (в нем неизвестные стоят поочередно — по пять на каждую точку области) и векторе правых частей **B**. Пример матрицы коэффициентов системы разностных уравнений приведен на рис. 10. Для визуализации разреженной матрицы в пакете MATLAB используется функция `spy(A)`.

Решение краевой задачи показано на рис. 11. Результаты расчета показывают, что сопряжение решения с граничным условием на бесконечности происходит при $x \sim 14$. До определенной высоты идет нарастание скорости по всем направлениям. Осевая составляющая скорости всюду положительна (везде происходит восходящее движение жидкости). Вблизи стенки радиальная составляющая скорости направлена внутрь (к оси вращения). На большой высоте восходящее течение затухает, так как там происходит радиальное движение жидкости наружу. Однако в целом в радиальном направлении преобладает движение жидкости внутрь (к оси вращения). Результаты расчетов согласуются с решением, приведенным в [9] в табличном виде, до третьего знака.

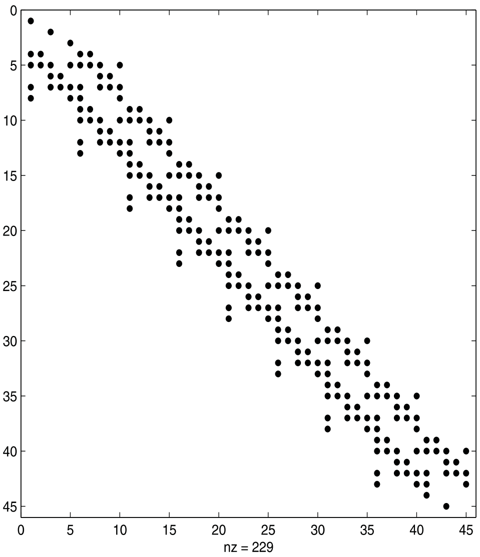


Рис. 10. Структура матрицы коэффициентов при $N = 9$. Матрица A размером 45×45 имеет 229 ненулевых элементов

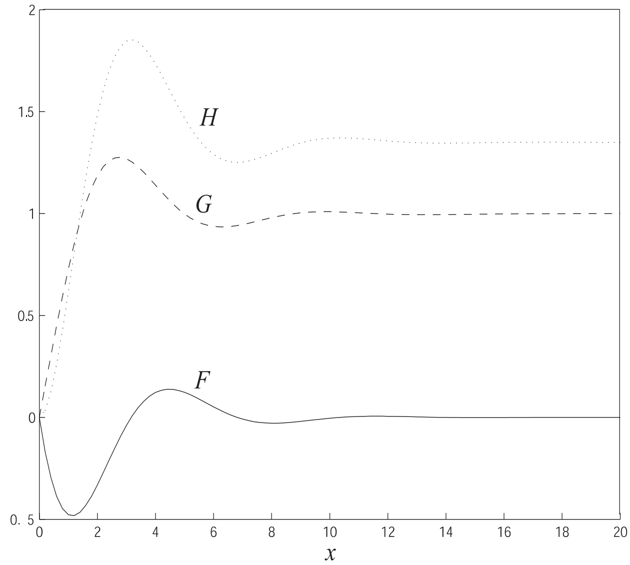


Рис. 11. Решение краевой задачи при $N = 101$ (F — радиальная скорость, G — окружная скорость, H — осевая скорость)

13. Реализация вычислительного алгоритма. Для реализации вычислительного алгоритма использовался гибкий набор способов взаимодействия MATLAB с внешними программами.

При использовании встроенного языка пакета для написания вычислительных алгоритмов затрачивается значительно меньше времени, чем на обычных языках программирования. Однако в тех случаях, когда составные части вычислительного алгоритма нельзя выразить при помощи векторных операторов MATLAB, приходится писать вложенные циклы и перераспределять память, что приводит к снижению скорости работы программы.

В связи с этим за основу бралась программа, написанная на языке C++. При переносе программы, написанной на языке C++, на язык MATLAB осуществлялась замена циклов по обработке одномерных и двумерных массивов соответствующими операторами и встроенными функциями пакета для обработки векторов и матриц. Для реализации расширения пакета создавалась dll библиотека при помощи среды Microsoft Visual C++ 6.0, которая подключалась к среде MATLAB. Код, написанный на языке C++, вызывался из пакета как обычная встроенная функция.

Использовался также и обратный подход, когда из программы, написанной на языке C++, вызывались операторы и библиотечные функции MATLAB, а полученный результат передавался в вызывающую программу (в этом случае MATLAB представляет собой вычислительный сервер для внешней программы).

Для повышения эффективности кодирования m-файлов использовался профилировщик, позволяющий определить время, затрачиваемое на выполнение каждой строки программы и выявить критические участки программы. Проводился как анализ правильности алгоритмов, решающих вспомогательные под-задачи, так и анализ правильности решения задачи в целом. Результаты расчетов сравнивались с имеющимися точными решениями (для многих автомодельных задач механики жидкости и газа такие решения приводятся в табличном виде).

14. Заключение. На основе разработанного подхода в пакете MATLAB реализовано решение различных краевых задач механики жидкости и газа. К числу таких задач относятся: задачи о течении жидкости в пограничном слое на плоской пластине и около критической точки (задача Блазиуса и задача Фолкнера–Скэн), в том числе с учетом неизотермичности течения и влияния магнитного поля; задача расчета течения в канале со вдувом; задачи о течении жидкости около тел произвольной геометрической формы при использовании переменных "функция тока–вихрь скорости" и криволинейных сеточных структур; задачи нестационарного нагрева пластины, цилиндра и шара лазерным импульсом, а также ряд других.

Программные разработки использованы на кафедре плазмогазодинамики и теплотехники Балтийского государственного технического университета "BOEHMEX" им. Д. Ф. Устинова при подготовке лабораторных практикумов по курсам "Численное моделирование в механике жидкости и газа", "Динамика вязкой жидкости", "Моделирование высокоинтенсивных процессов", "Математическое моделирование процессов в аэрокосмической технике", "Двухфазные течения".

СПИСОК ЛИТЕРАТУРЫ

1. *Амосов А.А., Дубинский Ю.А., Копченова Н.В.* Вычислительные методы для инженеров. М.: Высшая школа, 1994.
2. *Мэтьюз Д., Финк К.* Численные методы. Использование MATLAB. М.: Мир, 2001.
3. MATLAB 5.2. User's Guide. The Mathworks, Inc., 1998.
4. *Потемкин В.Г.* MATLAB 6: среда проектирования инженерных приложений. М.: ДИАЛОГ-МИФИ, 2003.
5. *Дьяконов В.П.* MATLAB 6/6.1/6.5 + Simulink 4/5 в математике и моделировании. М.: Солон-пресс, 2003.
6. MATLAB: Simulink & Toolboxes. M.: Softline, 1997.
7. MATLAB 5.2. Partial Differential Equation (PDE) Toolbox User's Guide. The Mathworks, Inc., 1998.
8. Численное решение многомерных задач газовой динамики / Под ред. С. К. Годунова. М.: Наука, 1976.
9. *Шлихтинг Г.* Теория пограничного слоя. М.: Наука, 1974.

Приложение

Программа решения краевой задачи для системы уравнений, описывающей вращательное движение жидкости над неподвижным основанием

Рассмотрим вопросы программной реализации численного решения краевой задачи для системы (17) в пакете MATLAB.

Численное решение задачи состоит в следующем.

1. Задается начальное приближение для вектора неизвестных $\mathbf{U}^n$ во всех точках расчетной области.

2. Получается новое приближение решения $\mathbf{U}_*^n$ путем построения матрицы A, вектора правых частей $\mathbf{B}$ и решения системы (14) из $5N$ линейных алгебраических уравнений (используется свойство разреженности матрицы коэффициентов).

3. Проверяется максимальная разность между решениями на двух последовательных итерациях $\mathbf{U}^n$ и $\mathbf{U}_*^n$ на предмет сходимости итерационного процесса. При достижении необходимой точности решения ($\varepsilon \sim 1\,\%$) процесс последовательных приближений завершается.

4. Для задания приближения на следующей итерации и ускорения сходимости итерационного процесса используется метод нижней релаксации

$$\mathbf{U}^{n+1} = (1-\omega)\,\mathbf{U}^n + \omega \mathbf{U}_*^n.$$

Коэффициент релаксации ω выбирается из интервала $0.1, \ldots, 0.4$ (такое значение подобрано опытным путем и обеспечивает достаточно высокую скорость сходимости итерационного процесса).

На подготовительном этапе производится задание входных данных, таких как размер расчетной области по пространственной координате `Ym`, число точек разностного шаблона `N` и их координат, шаг расчетной сетки `h`, коэффициент релаксации `omega`, погрешность расчета `Eps`, а также ряда вспомогательных данных.

Для хранения решения на итерациях n и $n+1$ вводятся два массива `UOld` и `UNew`, имеющих размерность `5N` (число уравнений в системе, умноженное на число расчетных точек). Правая часть системы хранится в массиве B размерности 5N. Для накопления погрешности и демонстрации сходимости итерационного процесса используется вспомогательный массив SErr.

Функция zeros(m,n) формирует массив нулей размерности m×n. Начальное приближение решения строится при помощи функции ones(m,n), которая формирует массив единиц размерности m×n. Оператор : используется для формирования векторов и матриц, а также для выделения из них подвекторов и подматриц.

```
% размер области по пространственной координате
Ym=20.0;

% шаг сетки
h=Ym/(N-1);

% вектор решения на итерациях n и n+1
UOld=zeros(N*5,1)';  UNew=UOld;

% начальное приближение решения
UOld((0:N-1)*5+1)=ones(N,1)-1;
UOld((0:N-1)*5+2)=ones(N,1)-1;
UOld((0:N-1)*5+3)=(0:h:1);
UOld((0:N-1)*5+4)=ones(N,1)-1;
UOld((0:N-1)*5+5)=ones(N,1)-1;

% вектор правых частей
B=zeros(N*5,1);

% погрешность на итерации
SErr=[];
```

Основную часть программы составляет итерационный цикл. Для проверки условия сходимости итерационного процесса вычисляется максимальная невязка по всем переменным и результат сравнивается с заданной величиной. При достижении требуемого уровня точности решения происходит выход из итерационного цикла, в противном случае тело цикла повторяется снова. Для задания приближения на новой итерации используется процедура нижней релаксации.

```
% итерационный цикл
Step=0;  Quit=1;
while (Quit)
 % число итераций
 Step=Step+1;

 % тело цикла
 ...

 % погрешность на итерации
 z=max(abs(UOld-UNew'));  SErr=[SErr z];
 % нижняя релаксация
 UOld=(1-beta)*UOld+beta*UNew';

 % условие выхода
 if (z<Eps|Step==50)
  Quit=0;
 end;
end
```

Производится последовательная нумерация всех интервалов между точками шаблона (всего их N−1) и уравнений системы. Для хранения номеров узловых точек и номеров уравнений используются массивы iP и iEq (по существу, в массиве iP заносятся номера узлов, лежащих посередине расчетного интервала, для которых записывается разностная схема).

```
% номера узлов
```

```
iP=1:N-1;
% номера уравнений
% (первые три уравнения выражают собой левые граничные условия)
iEq=(iP-1)*5+3;
```

В массивы iL и iR заносятся номера узлов, лежащих слева и справа от расчетного узла соответственно (для каждого уравнения системы).

```
% номера узлов слева  % номера узлов справа
iL1=(iP-1)*5+1;       iR1=iP*5+1;
iL2=(iP-1)*5+2;       iR2=iP*5+2;
iL3=(iP-1)*5+3;       iR3=iP*5+3;
iL4=(iP-1)*5+4;       iR4=iP*5+4;
iL5=(iP-1)*5+5;       iR5=iP*5+5;
```

В массив indI заносится номер уравнения, а в массив indJ — номер соответствующей неизвестной. В массиве sIJ хранятся коэффициенты перед неизвестными. Сначала производится постановка граничных условий на левой границе промежутка, в соответствующие массивы заносятся коэффициенты перед неизвестными и правая часть системы, затем задаются граничные условия на правой границе расчетного интервала.

```
% вспомогательные коэффициенты для построения разностной схемы
C1=0.5*h;
C2=0.25*h;

% вспомогательные массивы
UE=ones(N-1,1)';
Old1=UOld(iR1)+UOld(iL1);
Old2=UOld(iR2)+UOld(iL2);
Old3=UOld(iR3)+UOld(iL3);
Old4=UOld(iR4)+UOld(iL4);
Old5=UOld(iR5)+UOld(iL5);

% левые граничные условия
% для z1=F      % для z3=G      % для z5=H
indI=1;         indI=[indI 2];  indI=[indI 3];
indJ=1;         indJ=[indJ 3];  indJ=[indJ 5];
sIJ=1;          sIJ= [sIJ 1];   sIJ= [sIJ 1];
% правая часть
B(1)=0;         B(2)=0;         B(3)=0;

% первое уравнение
indI=[indI iEq+1 iEq+1 iEq+1 iEq+1];
indJ=[indJ iR1 iL1 iL2 iR2];
sIJ=[sIJ 1*UE -1*UE -C1*UE -C1*UE];
% правая часть
B(iEq+1)=ones(size(B(iEq+1),1),1)-1;

% второе уравнение
indI=[indI iEq+2 iEq+2 iEq+2 iEq+2 iEq+2 iEq+2 iEq+2 iEq+2];
indJ=[indJ iR2 iL2 iR5 iL5 iR3 iL3 iR1 iL1];
sIJ=[sIJ 1-C2*Old5 -1-C2*Old5 -C2*Old2 -C2*Old2 C2*2*Old3 ...
        C2*2*Old3 -C2*2*Old1 -C2*2*Old1];
% правая часть
B(iEq+2)=(h-C2*(Old2.*Old5-Old3.^2+Old1.^2)');

% третье уравнение
indI=[indI iEq+3 iEq+3 iEq+3 iEq+3];
indJ=[indJ iR3 iL3 iL4 iR4];
sIJ=[sIJ 1*UE -1*UE -C1*UE -C1*UE];
% правая часть
B(iEq+3)=ones(size(B(iEq+3),1),1)-1;
```

```
% четвертое уравнение
indI=[indI iEq+4 iEq+4 iEq+4 iEq+4 iEq+4 iEq+4 iEq+4 iEq+4];
indJ=[indJ iR4 iL4 iR3 iL3 iR1 iL1 iR5 iL5];
sIJ=[sIJ 1-C2*Old5 -1-C2*Old5 -C2*2*Old1 -C2*2*Old1 ...
        -C2*2*Old3 -C2*2*Old3 -C2*Old4 -C2*Old4];
% правая часть
B(iEq+4)=(-C2*(2*(Old3.*Old1)+Old4.*Old5)');

% пятое уравнение
indI=[indI iEq+5 iEq+5 iEq+5 iEq+5];
indJ=[indJ iR5 iL5 iR1 iL1];
sIJ=[sIJ 1*UE -1*UE 2*C1*UE 2*C1*UE];
% правая часть
B(iEq+5)=ones(size(B(iEq+5),1),1)-1;

% правые граничные условия
% для z1=F           % для z3=G
indI=[indI N*5-1];  indI=[indI N*5];
indJ=[indJ N*5-4];  indJ=[indJ N*5-2];
sIJ=[sIJ 1];        sIJ=[sIJ 1];
% правая часть
B(N*5-1)=0;         B(N*5)=1;
```

Сформированные данные используются для построения разреженной матрицы системы разностных уравнений. Функция `sparse(i,j,s)` формирует матрицу в соответствии с правилами записи разреженных матриц, принятыми в системе MATLAB. При вызове этой функции строки массива `[i j s]` используются для формирования разреженной матрицы размерности m×n, где `m=max(i)` и `n=max(j)`. Вектора `i` и `j` задают позиции элементов и являются целочисленными, а вектор `s` определяет числовое значение элемента матрицы. При формировании разреженной матрицы все строки вида `[i j 0]` из описания удаляются. Длина вектора `s` совпадает с количеством ненулевых элементов разреженной матрицы. Решение системы линейных уравнений реализуется в MATLAB с помощью специального монитора, который использует различные алгоритмы в зависимости от структуры матрицы. Эти алгоритмы реализованы в системе MATLAB на основе функций из пакета LINPACK.

```
% формирование разреженной матрицы
A=sparse(indI,indJ,sIJ);
% решение системы
UNew=A\B;
```

Для визуализации решения из вектора `UNew` выделяются функции F, G, H.

```
% поперечная координата
Y=(0:Ym/(N-1):Ym)';

% радиальная скорость
F=Uold((0:N-1)*5+1)';
% окружная скорость
G=Uold((0:N-1)*5+3)';
% осевая скорость
H=Uold((0:N-1)*5+5)';
```

После этого визуализация решения производится при помощи стандартных графических средств пакета MATLAB.

Поступила в редакцию
09.03.2004