УДК 681.3 (075)

И.В. Мельник

БАЗОВЫЕ ПРИНЦИПЫ МАТРИЧНОГО ПРОГРАММИРОВАНИЯ И ИХ РЕАЛИЗАЦИЯ В СИСТЕМЕ НАУЧНО-ТЕХНИЧЕСКИХ РАСЧЕТОВ MATLAB

Введение. В настоящее время при проведении сложных математических расчетов всё большее применение находят современные математические САПР. Это связано с наличием в них обширных библиотек математических функций, включающих элементарные, специальные функции, а также функции численных методов расчёта, с возможностями совмещения численных расчетов с аналитическими преобразованиями, а также с наличием современных средств программирования, используемых для решения нестандартных математических задач. Особое место среди математических САПР занимает система автоматизации научно-технических расчетов MatLab. Основной отличительной особенностью этой системы является её ориентация на использование матричных и векторных макроопераций при обработке числовых структур данных [1 – 4]. Другими важными преимуществами системы MatLab является открытый программный код и её ориентация на модульную архитектуру программирования. Система имеет свой язык программирования, ориентированный на решение сложных задач математического моделирования, что позволяет профессиональным программистам развивать и совершенствовать существующие математические библиотеки и создавать современные вычислительные комплексы. Ориентация системы MatLab именно на использование современных технологий программирования стала главной причиной популярности этой системы среди профессиональных программистов. Язык программирования системы MatLab достаточно прост, но разнообразен. В нём реализованы различные подходы, принципы и парадигмы современного программирования, включая структурное, объектно-ориентированное, логическое, функциональное, модульное программирование [1 – 4]. Особой популярностью пользуется эта система в академической и в университетской среде.

Однако при этом основным преимуществом системы MatLab, отличающим ей от традиционных языков программирования и соответствующих им программных средств, является развитый набор векторных и матричных макроопераций. Особенность этих операций состоит в том, что они позволяют обрабатывать структуры числовых данных как единое целое, без обращения к их отдельным элементам. Среди реализованных разработчиками системы матричных макроопераций чаще всего используются операции поэлементной обработки данных и множественная индексация массивов [1, 2]. Возможности использования матричных макроопераций настолько широкие, что на сегодняшний день они могут составить реальную альтернативу традиционным методам и средствам структурного программирования, описанным в [3 – 7]. Поэтому сегодня многие сторонники использования системы MatLab в инженерной и научной деятельности в своих исследованиях говорят о том, что при наличии развитого набора матричных макроопераций можно вообще отказаться от использования традиционных методов и средств структурного программирования, к которым относятся операторы ветвления и цикла. Несмотря на это средства матричного программирования в настоящее время изучены недостаточно. Анализ существующих на сегодняшний день базовых принципов матричного программирования проводится в данной статье.

Постановка задачи. Проведенные ранее исследования показали, что существующие в MatLab средства матричного программирования не являются полными, поскольку они не достаточны для решения ряда математических задач. В частности, они не позволяют реализовывать алгоритмы рекуррентных вычислений и многие функции численных методов без использования оператора цикла [3, 4, 6, 7]. Также было показано, что для реализации ветвящихся вычислительных процессов вместо условного оператора можно использовать арифметико-логические выражения [4, 9, 11], однако соответствующие этим выражениям средства матричного программирования до появления работ [9 – 11] в литературе не рассматривались. Поэтому для дальнейшего использования средств матричного программирования в научной и инженерной деятельности необходим тщательный анализ их возможностей и дальнейшее их развитие. Также были написаны специальные функции, предназначенные для обработки структур данных при организации рекуррентных вычислений [4, 9 – 11]. Формулирование базовых принципов матричного программирования, их систематизация, а также анализ возможностей их использования при решении прикладных задач программирования, являются целью данной статьи.

Арифметико-логические выражения и примеры их использования для решения стандартных задач программирования

Как отмечалось в работах [3, 4, 9 – 11], арифметико-логические выражения вида

$$F(x)=F_1(x)\cdot L_1(x)+F_2(x)\cdot L_2(x)+\ldots+F_n(x)\cdot L_n(x), \quad (1)$$

где $F_1(x)$, $F_2(x)$, …, $F_n(x)$ – арифметические функции, $L_1(x)$, $L_2(x)$, …, $L_n(x)$ – соответствующие им логические функции, могут составлять реальную альтернативу средствам структурного программирования и использоваться вместо ветвящихся структур. Любая кусочно-непрерывная функция, заданная на

соответствующих интервалах числовой оси, может быть записана в виде (1). Например, кусочно-непрерывной функции

$$\mathrm{f}(x)=\begin{cases}\dfrac{e^x}{x^2+1}, \mathrm{x}>0;\\ \dfrac{e^x-x^2}{x^2+1}, \mathrm{x}\le 0,\end{cases}$$

соответствует такое арифметико-логическое выражение [9]:

$$\mathrm{f}(x)=(x>0)\cdot\left(\frac{e^x}{x^2+1}\right)+(x\le 0)\cdot\left(\frac{e^x-x^2}{x^2+1}\right).$$

В работах [4, 9 – 11] было показано, что выражения вида (1) могут быть использованы для решения таких стандартных задач программирования, как поиск минимального или максимального элемента в выборке данных, сортировка элементов выборки по возрастанию или по убыванию, а также для подсчета количества элементов выборки, лежащих в заданном интервале. Например, для поиска максимального элемента в выборке может быть использовано такое выражение [4, 9 – 11]:

$$x_{\max}(1)=x(1);\ x_{\max}(i)=(x(i)\ge x_{\max}(i-1))\cdot x(i)+(x(i)<x_{\max}(i-1))\cdot x_{\max}(i-1). \qquad (2)$$

Выражения вида (2), где в арифметических и логических функциях используется индекс элемента *i*, называются рекуррентными арифметико-логическими выражениями. Такие выражения используются совместно с процедурами рекуррентных вычислений. Как отмечалось ранее, средства организации рекуррентных вычислений при обработке элементов векторов в базовой поставке пакета MatLab отсутствуют. Поэтому для организации рекуррентных вычислений над элементами векторов была написана специальная процедура recvect, программный код которой приведен в работе [4]. При использовании этой процедуры рекуррентная функция вида (2) задаётся в виде строковой переменной, и эта переменная является одним из параметров процедуры. Например, для расчета по формуле (2) можно использовать следующие командные строки системы MatLab.

Пример

```
» x=[1,100,-25,20,150,40,-50];
» ffh='v(ii)*(v(ii)>=w(ii-1))+w(ii-1)*(v(ii)<w(ii-1))';
» c=recvect(1,length(x),x,ffh,1e-10,1e10)
c =
   1  100  100  100  150  150  150
»
```

Таким образом, обычные арифметико-логические выражения вида (1) и рекуррентные арифметико-логические выражения вида (2) могут использоваться при написании программ в качестве реальной альтернативы операторам условного перехода. Аналогичным образом рекуррентные арифметико-логические выражения использовались для подсчета количества элементов выборки, лежащих в заданном интервале.

Выражения вида (1) применяются для представления кусочно-непрерывных функций, а выражения вида (2) могут быть использованы совместно с процедурой recvect для проведения рекуррентных вычислений. При этом удаётся полностью избежать использования операторов ветвления и цикла. При решении задач программирования представление ветвящихся вычислительных процессов в виде арифметико-логических выражений значительно более эффективно, чем использование сигма и дельта-функций или другие математические альтернативы [3, 4].

Рекуррентные матрицы и пример их использования

Более сложным, но и более эффективным способом организации рекуррентных вычислений через матричные макрооперации, является использование рекуррентных матриц. Сущность этого подхода состоит в том, что каждая следующая строка матрицы вычисляется через элементы её предыдущих строк. В общем случае соответствующую рекуррентную формулу можно записать в виде [4, 9 – 11]:

$$\begin{aligned}&\mathbf{M}_{<1>} = \mathbf{v_1}, \mathbf{M}_{<2>} = \mathbf{v_2}, ..., \mathbf{M}_{<n>} = \mathbf{v}_n,\\ &\mathbf{M}_{<i>} = f\left(i, i-1, i-2, ..., i-n, \mathbf{M}_{<i-1>}, \mathbf{M}_{<i-2>}, ..., \mathbf{M}_{<i-n>}\right),\end{aligned} \quad (3)$$

где **v** – векторы, $\mathbf{M}_{<i>}$ – строка матрицы с номером *i*.

Для вычисления элементов рекуррентной матрицы используются вектор-функции, позволяющие задать алгоритм расчёта для каждого отдельного элемента строки формируемой матрицы в виде вектора. Для формирования рекуррентных матриц была написана процедура recmat, код которой приведен в работе [4]. Вектор-функция при вызове процедуры recmat задаётся в виде строковой константы, также как и арифметико-логические выражения при вызове процедуры recvect. В качестве примера рассмотрим алгоритм пузырьковой сортировки элементов вектора, который через вектор-функцию может быть записан следующим образом [4, 9 – 11]:

$$\begin{pmatrix} xs_{i-1} \\ xs_i \end{pmatrix} = \begin{pmatrix} x_i > xs_{i-1} \\ x_i > xs_{i-1} \end{pmatrix} \cdot \begin{pmatrix} xs_{i-1} \\ x_i \end{pmatrix} + \begin{pmatrix} x_i \le xs_{i-1} \\ x_i \le xs_{i-1} \end{pmatrix} \cdot \begin{pmatrix} x_i \\ xs_{i-1} \end{pmatrix}, \quad (4)$$

где x – элементы исходного вектора, xs – элементы результирующего вектора. Пример использования выражения (4) для сортировки элементов приведен в работах [4, 9 – 11].

Использование рекуррентных матриц и векторов-функций для вычисления элементов их строк позволяет решить ряд задач математического моделирования, которые при использовании стандартных средств структурного программирования реализуются через циклы второго уровня вложенности [3 – 8]. Подробнее набор стандартных и оригинальных задач, которые удалось решить с использованием средств матричного программирования, будет рассмотрен в следующем разделе статьи.

Примеры стандартных и оригинальных задач, решенных с использованием средств матричного программирования системы MatLab

В первую очередь были рассмотрены возможности реализации с помощью средств матричного программирования задач численных методов. Поскольку большинство численных методов реализуются через рекуррентные алгоритмы, их удалось реализовать путём использования арифметико-логических выражений и процедуры recvect. В работах [4, 9 – 11] приведены программные коды, предназначенные для расчета степенных и функциональных рядов, для решения нелинейных уравнений и их систем методом простой итерации, для численных методов интегрирования функций, в частности, для метода трапеций и метода Симпсона, а также для метода Эйлера и метода Рунге-Кутта [12 – 15], используемых для решения дифференциальных уравнений и их систем. Для метода Рунге-Кутта в работе [14] была предложена контейнерная вычислительная схема.

Для более сложных вычислительных алгоритмов исследования были продолжены в работах [13, 14]. В частности, были построены вычислительные схемы для решения задач интерполяции и аппроксимации. Приведены коды программ для расчета коэффициентов интерполяционных полиномов Чебышева и для интерполяции функций кубическими сплайнами. Было показано, что использование арифметико-логических выражений позволяет представить функцию сплайн-интерполяции на всём интервале $[x_1; x_n]$ в следующем виде:

$$f(x) = \sum_{i=1}^{n} \left(\left(x \ge x_{i-1} \right) \& \left(x < x_i \right) \right) \cdot s_i(x), \quad (5)$$

где $s_i(x)$ – полином интерполяции на соответствующем отрезке $[x_{i-1}; x_i]$.

В работах [4, 11] также были проанализированы возможности использования средств матричного программирования для решения задач математической физики. В частности, с использованием рекуррентной матрицы и вектора-функции была решена задача о движении электрона в поле короткой магнитной линзы. В работе [11] приведен соответствующий программный код, в котором за счет использования процедуры recmat удалось полностью избежать использования операторов ветвления и цикла, которые считаются стандартными средствами структурного программирования при решении подобных вычислительных задач [6, 15]. Разработанные средства матричного программирования также были успешно использованы для решения задач анализа и обработки периодических и квазипериодических цифровых сигналов, в частности, для вычисления спектра сигнала и для расчета его автокорреляционной функции.

Алгоритмические свойств вычисляемых матриц и их анализ

В классической теории алгоритмов [5, 7] для анализа их особенностей и свойств обычно используются блок-схемы, которые не дают достаточной информации о вычислительных свойствах алгоритма, в частности, о возможности распараллеливания вычислительного процесса. Использование для вычислений рекуррентных матриц позволяет представить вычислительный процесс в виде

матричной диаграммы, аналогичной приведенной на рис. 1.

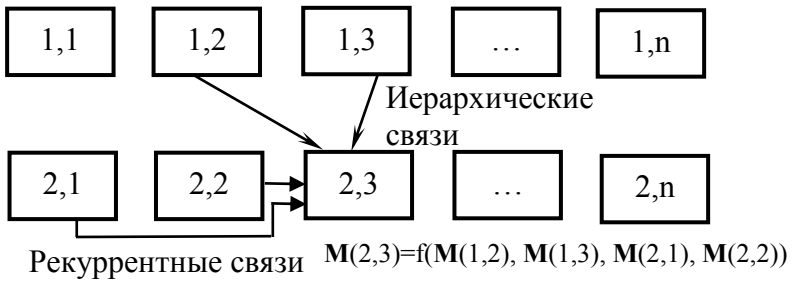

Рис. 1. Иллюстрация иерархических и рекуррентных связей в формируемой рекуррентной матрице

При построении матричной диаграммы указываются связи между элементами формируемой матрицы и функциональные зависимости между ними, при этом все связи разделяются на рекуррентные и иерархические. Иерархические связи характеризуют взаимосвязь вычисляемого элемента с элементами предыдущих строк, а рекуррентные связи – с ранее вычисленными элементами текущей строки. Различают также упорядоченные и неупорядоченные связи между элементами формируемой рекуррентной матрицы [14]. Процедура recmat написана таким образом, что позволяет при правильном задании вектора-функции учитывать все типы связей. Однако очевидно, что только те вычислительные процедуры, которые в рекуррентной матрице отображаются иерархическими связями, могут быть распараллелены. В работе [14] был предложен способ, который позволяет преобразовывать рекуррентные связи в иерархические путём замены элемента текущей строки на элемент того же столбца в предыдущей строке, однако такие преобразования не всегда допустимы. Для ускорения выполнения вычислительных процедур, отображаемых рекуррентными связями, могут использоваться контейнерные схемы [14].

При использовании средств матричного программирования можно предложить следующую методику написания программ [16].

1. Решение задачи математического моделирования в общем виде.

    1.1. Формирование обобщенной системы уравнений.

    1.2. Формирование на ее основе системы итерационных уравнений.

2. Формирование и анализ алгоритма вычислений, запись рекуррентной матрицы и вектор-функции. Анализ возможностей распараллеливания алгоритма.

    2.1. Формирование на основе полученной системы итерационных уравнений рекуррентной матрицы.

    2.2. Формирование вектор-функции, соответствующей полученной рекуррентной матрице.

    2.3. Анализ свойств рекуррентной матрицы и возможностей распараллеливания вычислительного процесса.

3. Написание программы.

    3.1. Определение и описание структур данных.

    3.2. Запись вектора-функции в виде строковой переменной.

    3.3. Решение полученного рекуррентного матричного уравнения с использованием процедуры recmat.

Выводы. Проведенные исследования позволяют сделать вывод о том, что, при использовании соответствующего инструментария и методов обработки данных, средства матричного программирования являются полными и достаточными и могут составить реальную альтернативу существующим средствам структурного программирования. При этом ветвящиеся вычислительные процессы реализуются с помощью арифметико-логических выражений, рекуррентные вычисления – с помощью дополнительно написанных процедур, а функция, описывающая вычислительный алгоритм, задаётся в виде строковой переменной. Предлагаемая методика программирования может найти применение в системе образования при изучении теории алгоритмов, а также в академической среде при разработке и тестировании новых вычислительных методов. Проведенные тестовые эксперименты показали, что с использованием разработанных средств матричного программирования можно реализовать

очень большое количество классических и оригинальных алгоритмов. Отличительными особенностями предлагаемой методики матричного программирования являются её относительная простота, а также возможность анализа результатов вычислений и поиска ошибок программирования на всех этапах вычислительного процесса.

ЛИТЕРАТУРА

1. Дьяконов В.П. Matlab 6/6.1/6.5 + Simulink 4/5. Полное руководство пользователя. – М.: Солон-Пресс, 2002. – 768 с.
2. Мартынов Н.Н. Введення в Matlab 6. – М.: Кудиц-Образ, 2002. – 388 с.
3. Мельник І.В. Система науково-технічних розрахунків MatLab та її використання для розв'язання задач із електроніки: навчальний посібник у 2-х томах. Т. 1. Основи роботи та функції системи. – К.: Університет «Україна», 2009. – 507 с.
4. Мельник І.В. Система науково-технічних розрахунків MatLab та її використання для розв'язання задач із електроніки: навчальний посібник у 2-х томах. Т. 2. Основи програмування та розв'язання прикладних задач. – К.: Університет «Україна», 2009. – 327 с.
5. Сэджвик Р. Фундаментальне алгоритмы на С++. Анализ. Структуры данных. Сортировка. Поиск. Пер с англ. – СПб., ООО „ДиаСофт ЮП", 2002. – 688 с.
6. Мэтьюз Д., Куртис Д. Численные методы. Использование MatLab. – М.: Издательский дом «Вильямс», 2001. – 720 с.
7. Кнудт Д. Искусство программирования. / перевод с англ. под общей редакцией Ю.В. Козаченко. – М.: Издательский дом «Вильямс», 2003. – 560 с.
8. Самарский А.А., Гулин А.В. Численные методы: учебное пособие для вузов. – М.: Наука, 1989. – 432 с.
9. Мельник И.В. Использование матричных макроопераций при работе в математических САПР на примере системы MatLab. – Вестник Херсонского национального технического университета. Вып. 2 (28). – Херсон, 2007. – С. 199-205.
10. Мельник И.В. Анализ возможностей использования матричных макроопераций системы MatLab при решении физических задач // Сборник трудов конференции «Моделирование – 2008». – Институт проблем моделирования в энергетике им. Г.Е. Пухова НАН Украины. – С. 453–460.
11. Мельник И.В. Анализ возможностей использования матричных макроопераций системы MatLab при решении прикладных задач // Электронное моделирование. – №3. – 2009. – С. 37-51.
12. Мельник И.В., Шинкаренко Н.В. Анализ взаимосвязей вычисляемых элементов матриц при реализации рекуррентных алгоритмов с помощью матричных макроопераций. – Моделированне и информационные технологии. Материалы международной научной конференции „Моделирование – 2010", 12-14 мая 2010 г. – Киев, 2010. – С. 253 – 261.
13. Мельник И.В., Шинкаренко Н.В. Реализация вычислительных алгоритмов среднего уровня сложности в системе MatLab с использованием матричных макроопераций // Вестник Херсонского национального технического университета. Вып. 3 (39). – Херсон, 2010. – С. 322 – 327.
14. Мельник И.В., Шинкаренко Н.В. Анализ алгоритмических особенностей вычисляемых матриц при решении задач программирования средствами матричных макроопераций. // Электронное моделирование. – Т. 33. – №2. – 2011. – С. 81-92.
15. Ильина В.А., Силаева П.К. Численные методы для физиков-теоретиков. – Москва-Ижевск: Институт компьютерных исследований, 2003. – 132 с.
16. Мельник И.В. Исследование возможностей использования матричных макроопераций при решении стандартных задач программирования. // Сборник трудов конференции «Моделирование-2012», 16-18 мая 2012 г. – Киев, 2012. – С. 277–280.

МЕЛЬНИК Игорь Витальевич – доктор техн. наук, доцент кафедры электронных приборов и устройств факультета электроники Национального Технического Университета Украины "Киевский политехнический институт".

Научные интересы:

– компьютерное моделирование электронно-лучевых устройств и технологических процессов на их основе, современные методы и средства математического моделирования.