УДК 519.612.2

ПРЯМОЙ МЕТОД РЕШЕНИЯ СИСТЕМЫ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ УРАВНЕНИЙ НА ОСНОВЕ ВЕЙВЛЕТ-РАЗЛОЖЕНИЯ

О. В. Филиппов

Рассматривается прямой метод решения СЛАУ, в основе которого лежит нестандартное представление в вейвлет-базисе матрицы системы. Дается краткое введение в вейвлеты и описывается нестандартное представление оператора в вейвлет-базисе. Обсуждается, почему прямой метод при использовании вейвлетов становится эффективным, даже когда рассматриваются плотнозаполненные матрицы.

Введение

Существует много доступных методов решения больших сильноразреженных систем (CPG) линейных уравнений. Наиболее популярными методами решения для таких систем являются итерационные методы.

Прямые методы решения СРС могут также использоваться для решения таких систем. Их эффективность зависит от числа дополнительных ненулевых членов, полученных в ходе LU-факторизации матрицы системы. Напомним, что некоторые разреженные системы не имеют разреженных LU-факторов. В работе [2] предложен метод построения разреженного представления матриц, которые изначально плотно-заполнены.

Далее будет показано, что, если использовать нестандартное представление в вейвлет-базисе матрицы системы, соответствующие LU-факторы будут также сильноразреженными, это делает эффективным прямой метод решения СЛАУ.

Перечислим основные определения и свойства вейвлетов.

Вейвлет-функция ψ строится по масштабирующей функции φ, которая обладает двумя свойствами:

а) система функций $\{\varphi(x-k), k \in Z\}$ ортогональна в $L^2(R)$,

б) функция является решением функционального уравнения

$$\varphi(x) = \sqrt{2}\sum_k h_k \varphi(2x-k). \tag{1}$$

Тогда для вейвлет-функции ψ верно представление

$$\psi(x) = \sqrt{2}\sum_k (-1)^k h_{1-k}\varphi(2x-k). \tag{2}$$

Наибольший интерес представляют компактно-сосредоточенные функции φ и ψ. В этом случае фильтр h_k конечен.

Пусть $V_0 = span\{\varphi(x-k), k \in Z\} = \left\{f \in L^2(R), f(x) = \sum c_k \varphi(x-k), \sum c_k \le \infty\right\}$ и $\varphi_{J,k}(x) = 2^{J/2}\varphi(2^J x - k)$. Через V_J обозначим линейную оболочку $span\left\{2^{J/2}\varphi(2^J - k), k \in Z\right\}$.

Из соотношения (1) следует, что

$$\ldots \subset V_{-1} \subset V_0 \subset V_1 \subset \ldots$$

Определим подпространство W_J как ортогональное дополнение V_J в V_{J+1}, т.е. $V_{J+1} = V_J \oplus W_J$. Если $\overline{\bigcup_J V_J} = L^2(R)$ и $\bigcap_J V_J = \{0\}$, то доказывается в [1], что

$$L^2(R) = V_0 \oplus_{J \ge 0} W_J.$$

Система функций $\{\psi_{j,k}(x)\}_{j\geq 0}$ и $\{\varphi_{0,k}(x)\}$ является ортогональным базисом в пространстве $L^2(R)$. В частности, $V_n = V_0 \oplus_{0\leq j\leq n-1} W_j$.

Тогда система функций $\{\psi^{пер}_{j,k}(x)\}_{j\geq 0}$ и $\{\varphi^{пер}_{0,k}(x)\}$, где

$$\psi^{пер}_{j,k}(x) = \sum_l \psi_{j,k}(x+l) \text{ и } \varphi^{пер}_{0,k}(x) = \sum_l \varphi_{0,k}(x+l),$$

является ортогональным базисом в $L^2(0,1)$. Определим проектор P_j на подпространство V_j, $j \in Z$,

$$P_j : L^2(R) \to V_j,$$

и Q_j на подпространстве W_j

$$Q_j : L^2(R) \to W_j,$$

где $Q_j = P_{j-1} - P_j$. Для линейного оператора $T : L^2(0,1) \to L^2(0,1)$ оператор $T_j = P_j T P_j$ будет называться дискретизацией оператора T масштаба j. В базисе $\varphi^l_{j,k}(x,y)$ с оператором T_0 можно связать матрицу $\mathcal{T}_0$ размера $2^j \times 2^j$.

Введем ортогональную матрицу

$$\mathcal{W}_{j-1} = \begin{bmatrix} \mathcal{Q}_{j-1} \\ \mathcal{P}_{j-1} \end{bmatrix}, \tag{3}$$

где $\mathcal{P}_{j-1}$ и $\mathcal{Q}_{j-1}$ – матричное представление проекторов P_{j-1} и Q_{j-1} соответственно. Пусть $x_j = \mathcal{P}_j x$ и $y_j = \mathcal{P}_j y$ – дискретизации элементов x и y из $L^2(0,1)$. Для того, чтобы свести

$$\mathcal{T}_j x_j = y_j \tag{4}$$

к масштабу $j-1$, применим (3) и (4)

$$(\mathcal{W}_{j-1}\mathcal{T}_j\mathcal{W}_{j-1})(\mathcal{W}_{j-1}x_j) = (\mathcal{W}_{j-1}y_j). \tag{5}$$

Получим

$$\begin{bmatrix} \mathcal{A}_{j-1} & \mathcal{B}_{j-1} \\ \mathcal{C}_{j-1} & \mathcal{T}_{j-1} \end{bmatrix} \begin{bmatrix} \tilde{d}_{j-1} \\ \tilde{s}_{j-1} \end{bmatrix} = \begin{bmatrix} d_j \\ s_j \end{bmatrix}, \tag{6}$$

где $d_j = \mathcal{Q}_j y$, $s_j = \mathcal{P}_j y$, $\tilde{d}_{j-1} = \mathcal{Q}_{j-1} x$, $\tilde{s}_{j-1} = \mathcal{P}_{j-1} x$, в то время как $\mathcal{A}_{j-1}$, $\mathcal{B}_{j-1}$ и $\mathcal{C}_{j-1}$ – матричные операторы

$$\begin{aligned} A_{j-1} &: W_{j-1} \to W_{j-1} \quad \mathcal{A}_{j-1} = \mathcal{Q}_{j-1}\mathcal{T}\mathcal{Q}_{j-1} \\ B_{j-1} &: V_{j-1} \to W_{j-1} \quad \mathcal{B}_{j-1} = \mathcal{Q}_{j-1}\mathcal{T}\mathcal{P}_{j-1} \\ C_{j-1} &: W_{j-1} \to V_{j-1} \quad \mathcal{C}_{j-1} = \mathcal{P}_{j-1}\mathcal{T}\mathcal{Q}_{j-1} \\ T_{j-1} &: V_{j-1} \to V_{j-1} \quad \mathcal{T}_{j-1} = \mathcal{P}_{j-1}\mathcal{T}\mathcal{P}_{j-1} \end{aligned}$$

Матрицы $\mathcal{A}_{j-1}$, $\mathcal{B}_{j-1}$ и $\mathcal{C}_{j-1}$ содержат уточняющую информацию об операторе T_{j-1}. На следующем шаге матрица $\mathcal{T}_{j-1}$ извлекается из (6), и к ней применяется ортогональная матрица $\mathcal{W}_{j-2}$, как и в (5). Эта процедура продолжается на всех масштабах $j = n-1,\ldots,0$. Результатом является нестандартная форма (NS-форма) исходной матрицы $\mathcal{T}_0$. Для $j_0 = 3$ NS-форма матрицы $\mathcal{T}_0$ показана на рис. 1. Здесь $\tilde{d}_j, \tilde{s}_j$ и $\breve{d}_j, \breve{s}_j$ связаны соотношениями

$$\begin{aligned} \breve{d}_j &= \mathcal{A}_j\tilde{d}_j + \mathcal{B}_j\tilde{s}_j, \\ \breve{s}_j &= \mathcal{C}_j\tilde{d}_j, \; j = j_0 - 1,\ldots,0, \end{aligned}$$

и на последнем шаге

$$\breve{s}_0 = C_0\tilde{d}_0 + T_0\tilde{s}_0.$$

Для того, чтобы конвертировать $\left\{\breve{d}_j, \breve{s}_j\right\}_{0\le j\le J_0-1}$ в вейвлет-представление $\left\{\left\{d_j\right\}_{0\le j\le J_0-1}, s_0\right\}$, следует воспользоваться алгоритмом:

1. $\bar{d}_{J_0-1} = 0, \bar{s}_{J_0-1} = 0.$
2. $\bar{d}_j = Q_j\left(\breve{s}_{j-1} + \bar{s}_{j-1}\right), \bar{s}_j = P_j\left(\breve{s}_{j-1} + \bar{s}_{j-1}\right).$
3. Тогда $d_j = \breve{d}_j + \bar{d}_j$, а на последнем шаге $s_0 = \breve{s}_0 + \bar{s}_0$.

Пусть система функций $\psi_k(t)$ ортогональна и

$$x(t) = \sum_{k=-\infty}^{\infty} c_k\psi_k(t),$$

где коэффициенты c_k вычисляются по функции

$$c_k = \int f(t)\psi_k(t)dt.$$

Если $\psi_k(t)$ имеют нулевые моменты

$$0 = \int t^m\psi_k(t)dt, 0 \le m \le n-1,$$

тогда функция f хорошо аппроксимируется многочленом $\le n-1$. Коэффициенты разложения будут малыми или нулевыми. Поэтому вейвлеты, имеющие нулевые моменты, могут использоваться для «сжатия» гладких данных.

Если T является интегральным оператором и его ядро - гладкая функция вне диагонали, то матрица, полученная «дискретизацией», имеет много малых элементов вне диагонали. Такие матрицы после «сжатия» имеют разреженное представление в вейвлет-базисе. Поэтому NS-форма матрицы после «сжатия» становится сильно разреженной.

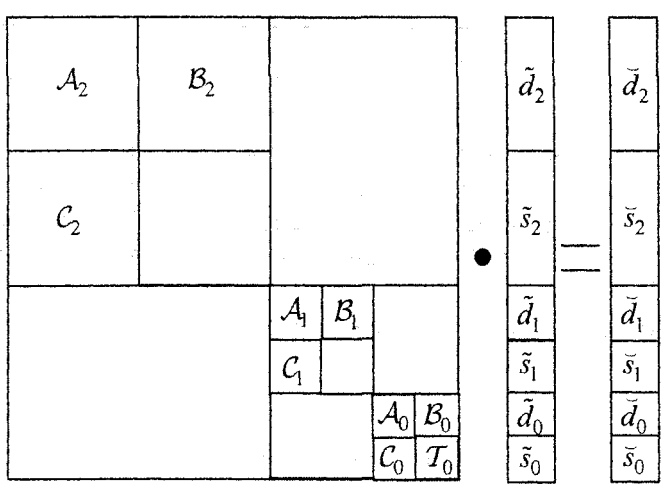

Рис. 1. Матричное уравнение в нестандартном представлении.

Алгоритм

Рассмотрим алгоритм решения СЛУ методом нестандартного представления в вейвлет-базисе. Его общий вид можно увидеть на рис. 2.

Рис. 2. Алгоритм

Прокомментируем шаг 2. Выше уже описывался способ построения нестандартной формы матрицы, но для данного алгоритма предлагается совместить его с LU-разложением (более подробное описание данного подхода можно увидеть в работе [2]). Результирующая процедура получится более эффективной, чем если бы нестандартное и LU-разложения применялись независимо.

В начале расчитаем вейвлет-разложение оператора T_n для «первого» масштаба:

$$T_n = A_{n-1} + B_{n-1} + C_{n-1} + T_{n-1},$$

где

$$A_{n-1} = Q_{n-1} T_n Q_{n-1},$$
$$B_{n-1} = Q_{n-1} T_n P_{n-1},$$
$$C_{n-1} = P_{n-1} T_n Q_{n-1},$$
$$T_{n-1} = P_{n-1} T_n P_{n-1}.$$

Используя стандартное LU-разложение, получим:

$$A_{n-1} = L_{n-1} U_{n-1},$$

и зададим $\hat{A}_{n-1} = L_{n-1}$ и $\tilde{A}_{n-1} = U_{n-1}$. Имея $\hat{A}_{n-1}$ и $\tilde{A}_{n-1}$, можно получить $\tilde{B}_{n-1}$ и $\hat{C}_{n-1}$ решая

$$\hat{A}_{n-1} \tilde{B}_{n-1} = B_{n-1},$$
$$\hat{C}_{n-1} \tilde{A}_{n-1} = C_{n-1}$$

с помощью стандартных методов прямого и обратного прохода.

Для продолжения на следующем масштабе необходимо вейвлет-разложение оператора T_{n-1} и $\hat{C}_{n-1}\tilde{B}_{n-1}$. Эти процедуры могут быть скомбинированы, если вначале вычесть $\hat{C}_{n-1}\tilde{B}_{n-1}$ из T_{n-1}, а затем рассчитать

$$T_{n-1}-\hat{C}_{n-1}\tilde{B}_{n-1}=\left(A_{n-2}-\bar{A}_{n-2}\right)+\left(B_{n-2}-\bar{B}_{n-2}\right)+\left(C_{n-2}-\bar{C}_{n-2}\right)+\left(T_{n-2}-\bar{T}_{n-2}\right), \quad (7)$$

где

$$A_{n-2}-\bar{A}_{n-2}=Q_{n-2}\left(T_{n-1}-\hat{C}_{n-1}\tilde{B}_{n-1}\right)Q_{n-2},$$

$$B_{n-2}-\bar{B}_{n-2}=Q_{n-2}\left(T_{n-1}-\hat{C}_{n-1}\tilde{B}_{n-1}\right)P_{n-2},$$

$$C_{n-2}-\bar{C}_{n-2}=P_{n-2}\left(T_{n-1}-\hat{C}_{n-1}\tilde{B}_{n-1}\right)Q_{n-2},$$

$$T_{n-2}-\bar{T}_{n-2}=P_{n-2}\left(T_{n-1}-\hat{C}_{n-1}\tilde{B}_{n-1}\right)P_{n-2}.$$

Используя стандартное LU-разложение, получим

$$A_{n-2}-\bar{A}_{n-2}=L_{n-2}U_{n-2},$$

где $\hat{A}_{n-2}=L_{n-2}$ и $\tilde{A}_{n-2}=U_{n-2}$. Имея $\hat{A}_{n-2}$ и $\tilde{A}_{n-2}$, можно получить $\tilde{B}_{n-2}$ и $\hat{C}_{n-2}$ решая

$$\hat{A}_{n-2}\tilde{B}_{n-2}=B_{n-2}-\bar{B}_{n-2},$$

$$\hat{C}_{n-2}\tilde{A}_{n-2}=C_{n-2}-\bar{C}_{n-2},$$

с помощью стандартных методов прямого и обратного прохода.

Для продолжения необходимо рассчитать вейвлет-разложение T_{n-2}, $\hat{C}_{n-2}\tilde{B}_{n-2}$ и дополнительного члена $\bar{T}_{n-2}$ из равенства (7).

$$T_{n-2}-\hat{C}_{n-2}\tilde{B}_{n-2}-\bar{T}_{n-2}=\left(A_{n-3}-\bar{A}_{n-3}\right)+\left(B_{n-3}-\bar{B}_{n-3}\right)+\left(C_{n-3}-\bar{C}_{n-3}\right)+\left(T_{n-3}-\bar{T}_{n-3}\right),$$

где

$$A_{n-3}-\bar{A}_{n-3}=Q_{n-3}\left(T_{n-2}-\hat{C}_{n-2}\tilde{B}_{n-2}-\bar{T}_{n-2}\right)Q_{n-3},$$

$$B_{n-3}-\bar{B}_{n-3}=Q_{n-3}\left(T_{n-2}-\hat{C}_{n-2}\tilde{B}_{n-2}-\bar{T}_{n-2}\right)P_{n-3},$$

$$C_{n-3}-\bar{C}_{n-3}=P_{n-3}\left(T_{n-2}-\hat{C}_{n-2}\tilde{B}_{n-2}-\bar{T}_{n-2}\right)Q_{n-3},$$

$$T_{n-3}-\bar{T}_{n-3}=P_{n-3}\left(T_{n-2}-\hat{C}_{n-2}\tilde{B}_{n-2}-\bar{T}_{n-2}\right)P_{n-3}.$$

Процесс продолжается до масштаба n, где рассчитываются члены $\hat{T}_0$ и $\tilde{T}_0$.

Программа

На основе вышеприведенного алгоритма нами была разработана программа[1] для решения СЛАУ. В качестве языка программирования был выбран Fortran90, для работы с сильно разреженными матрицами была использована библиотека SPARSKIT2.

Основную работу выполняет процедура *solveUsingNSForm,* на вход которой подается матрица коэффициентов, вектор свободного члена, векторы коэффициентов материнской и вейвлет-функций, количество вейвлет-преобразований, которое необходимо будет проделать над матрицей коэффициентов в процессе получения нестандартной формы, и переменная, в которой будет возвращен вектор неизвестных. Стоит отметить, что все матрицы и векторы передаются в сжатом формате (так называемый, формат CSR).

Работу процедуры *solveUsingNSForm* можно разделить на 5 частей.

1. *Расчет нестандартного представления для матрицы коэффициентов.* Это работу проделывает процедура *createNSForm2Dim.* Ей на вход подается матрица, которую необходимо преобразовать, векторы коэффициентов материнской и вейвлет-функций, количество преобразований и переменные, в которых будут возвращены верхнетреугольное

[1] В момент написания данной статьи программа проходила процесс регистрации в ФГНУ «Государственном координационном центре информационных технологий» Федерального агентства по образованию.

и нижнетреугольное нестандартные представления исходной матрицы. Данная процедура выполняется, четко следуя шагу 2 вышеприведенного алгоритма.

2. *Расчет нестандартного представления для вектора свободного члена.* Для этого используется процедура *createNSFormlDim*. Ей на вход подается вектор свободного члена, векторы коэффициентов материнской и вейвлет-функций, количество преобразований и переменная, в которой будет возвращен результат - нестандартная форма вектора. В процессе выполнения данной процедуры исходный вектор претерпевает заданное количество вейвлет-преобразований, причем получающиеся векторы запоминаются в переменную-результат.

3. *Решение первого уравнения (см. шаг 5 алгоритма) в нестандартной форме методом прямого прохода.* Для этого используется процедура *nsForwardSubstitution*. На вход подается нижнетреугольная нестандартная форма, вектор свободного члена в нестандартном представлении, векторы коэффициентов материнской и вейвлет-функций и переменная, в которой вернется результат решения.

4. *Решение второго уравнения (см. шаг 6 алгоритма) в нестандартной форме методом обратного прохода.* Эту работу выполняет процедура *nsBackwardSubstitution*. На вход подается верхнетреугольная нестандартная форма, результат, полученный на предыдущем шаге, векторы коэффициентов материнской и вейвлет-функций, переменная, в которой вернется результат решения.

5. *Обратное вейвлет-преобразование для получения вектора решения в стандартной форме.* Для этого используется процедура *inverseWaveletTranslDim,* на вход которой подается вектор материнских и вейвлет-коэффициентов первого масштаба искомого вектора, векторы коэффициентов материнской и вейвлет-функций, переменная, в которой вернется результирующий вектор.

Для проведения всех стандартных действий над матрицами и векторами используются процедуры и функции из библиотеки SPARSKIT2. Она содержит процедуры для преобразования матриц из стандартного формата в «сжатый» и обратно, решения СЛАУ методами прямого и обратного прохода, LU-разложения матрицы, умножения и сложения матриц и так далее.

Численные результаты

В этом разделе представлены численные результаты обработки нескольких примеров для демонстрации быстродействия и точности алгоритма и программы. Для тестирования был выбран компьютер с процессором Intel(R) Pentium(R) 4 CPU 3,00GHz с 2048Mb оперативной памяти.

Пример 1. Для первого примера была выбрана матрица вида:

$$A_{i,j} = \begin{cases} C \Big/ \tan\left(\pi(i-j)\Big/N\right), & i \neq j, \\ 1, & i = j, \end{cases}$$

где $N = 2^n$ – размерность матрицы, $i, j = 1..2^n$ и $C = 1/2^n$. Правая часть матричного уравнения подбиралась так, чтобы существовало известное точно решение.

Данная СЛАУ была решена при помощи нестандартного представления и при помощи библиотеки *Lapack*. Из этого пакета была выбрана процедура *DSGESV,* реализующая один из наиболее часто используемых итерационных алгоритмов решения СЛАУ. Более подробно об этом алгоритме можно узнать из документации, расположенной по адресу *http://www.netlib.org/lapack*.

Результаты можно увидеть в табл. 1.

Таблица 1

N	T_{NS}	$L_\infty(NS)$	$L_2(NS)$	T_{Lapack}	$L_\infty(Lapack)$	$L_2(Lapack)$
2^9	0,68	$1,74\cdot10^{-4}$	$4,59\cdot10^{-4}$	0,78	$3,11\cdot10^{-15}$	$1,36\cdot10^{-14}$
2^{10}	3,35	$4,25\cdot10^{-4}$	$1,19\cdot10^{-3}$	6,18	$3,33\cdot10^{-15}$	$2,48\cdot10^{-14}$
2^{11}	19,32	$6,53\cdot10^{-4}$	$2,25\cdot10^{-3}$	48,18	$9,05\cdot10^{-15}$	$8,39\cdot10^{-14}$

В первом столбце этой таблицы находится размерность обрабатываемой матрицы, во втором время в секундах работы алгоритма на основе нестандартного представления, в третьем и четвер-

том ошибка данного алгоритма, в пятом время работы в секундах процедуры из библиотеки *Lapack,* в двух оставшихся - ошибка этой процедуры.

Пример 2. Во втором примере использовалась матрица вида:

$$A_{i,j} = \begin{cases} \sin(\pi(i-j)/N), & i \neq j \\ 1, & i = j \end{cases},$$

где $N = 2^n$ - размерность матрицы и $i, j = 1..2^n$. Правая часть данного матричного уравнения подбиралась так, чтобы существовало известное точно решение.

Результаты можно увидеть в табл. 2, построенной аналогично табл. 1.

Таблица 2

N	T_{NS}	$L_\infty(NS)$	$L_2(NS)$	T_{Lapack}	$L_\infty(Lapack)$	$L_2(Lapack)$
2^9	0,73	$1{,}46\cdot10^{-4}$	$1{,}01\cdot10^{-3}$	0,78	$5{,}81\cdot10^{-13}$	$3{,}73\cdot10^{-12}$
2^{10}	1,83	$1{,}58\cdot10^{-3}$	$3{,}54\cdot10^{-2}$	6,15	$1{,}54\cdot10^{-12}$	$1{,}24\cdot10^{-11}$
2^{11}	8,14	$1{,}47\cdot10^{-3}$	$2{,}81\cdot10^{-2}$	48,59	$4{,}17\cdot10^{-12}$	$4{,}91\cdot10^{-11}$

Здесь стоит отметить, что, хотя алгоритм на основе нестандартного представления работает и быстрее, чем процедура из библиотеки *Lapack,* но точность полученного решения у него существенно ниже. Снижение точности происходит вследствие трешолдинга, производимого в момент вейвлет-преобразования и LU-разложения. Суть трешолдинга состоит в том, что величины меньше некоего заданного порога считаются равными нулю. Все предыдущие примеры обрабатывались с порогом равным 10^{-5} . В табл. 3 можно увидеть, как будет меняться точность и время решения, если уменьшать этот порог.

Таблица 3

Порог	T_{NS}	$L_\infty(NS)$	$L_2(NS)$
10^{-8}	43,55	$1{,}51\cdot10^{-8}$	$7{,}75\cdot10^{-8}$
10^{-9}	44,38	$1{,}29\cdot10^{-9}$	$2{,}13\cdot10^{-9}$
10^{-11}	48,01	$6{,}78\cdot10^{-12}$	$7{,}84\cdot10^{-11}$

Таким образом, видно, что для порога равного 10^{-11}, ошибка решения становится одного порядка с ошибкой процедуры *Lapack.* Что касается времени выполнения, то оно также становится сравнимым. Однако, стоит взглянуть что произойдет на следующей размерности матрицы (табл. 4).

Таблица 4

N	T_{NS}	$L_\infty(NS)$	$L_2(NS)$	T_{Lapack}	$L_\infty(Lapack)$	$L_2(Lapack)$
2^{12}	362,87	$1{,}31\cdot10^{-10}$	$5{,}01\cdot10^{-10}$	387,36	$1{,}46\cdot10^{-11}$	$2{,}03\cdot10^{-10}$

Время работы алгоритма на основе нестандартного представления сопоставимо и даже несколько меньше, чем время работы итерационного алгоритма при сравнимой ошибке. Стоит отметить, что для вейвлет-преобразований, подбирая вейвлет-функцию, можно добиться существенно-большего числа нулей в результате, а значит, увеличить общую скорость. Для данных примеров использовалась вейвлет-функция D4 (Добеши 4 порядка).

Итак, можно сделать вывод о том, что алгоритм решения СЛАУ на основе нестандартного представления работает сравнимо по времени выполнения и по ошибке результата с итерационными алгоритмами.

Литература

1. Добеши, И. Десять лекций по вейвлетам / И. Добеши. - Ижевск: НИЦ «Регулярная и хаотическая динамика». - 2001. - 464 с.

2. David L. Gines LU Factorization of Non-Standard Forms and Direct Multiresolution Solver / David L. Gines, G. Beylkin, J. Dunn. - Appl. Comput. Harmon. Anal. - 1998. -№ 5(2). - P. 156-201.

Поступила в редакцию 9 июня 2007 г.