матрица, заключенная в скобки, есть не что иное, как одна из матриц $F_1, \dots, F_M$. Следовательно, матрица LP_{ij} имеет строки, совпадающие с какими-то строками матрицы L либо нулевые, т.е. $LP_{ij} = Q_{ij} L$, где Q_{ij} в каждой строке имеет только нули и, возможно, одну единицу. Понятно, что умножение матрицы Q_{ij} на вектор выполняется с максимальной параллельностью. Вводя дополнительные линейные формы, определяемые строками матрицы Q_{ij}, мы получаем T-алгоритм, удовлетворяющий условиям теоремы 6.2.3. Значит, векторный алгоритм с требуемыми свойствами существует. Теорема доказана.

Свойство алгоритма быть T-алгоритмом обычно обнаруживается без особого труда. Однако конкретный вид векторов $q_i^{(k)}$ и линейных форм от их компонент определяется неоднозначно. В нашем методе векторизации мы отделяем вычисление линейных форм от вычисления векторов $q_i^{(k)}$. С учетом этого бывает полезно провести расщепление множества линейных форм на классы и строить параллельные алгоритмы для каждого из классов в отдельности. В ряде случаев удается получить классы, позволяющие применить теоремы 6.2.2-6.2.4.

§ 7. Векторные алгоритмы для матриц типа тёплицевых

7.I. Тёплицевы матрицы

Рассмотрим алгоритм (5.2.5). Его максимальная параллельная форма содержит $O(n^2)$ ярусов, т.е. алгоритм практически не распараллеливается. Однако очевидно, что это есть T-алгоритм. Множество всех линейных форм естественным образом разбивается на два класса: $\{F_k\}$ и $\{G_k\}$.

Введем следующие $n-1$-мерные векторы:

$$q_1^{(k)} = [0 \dots 0 \; x_0^{(k)} \dots x_k^{(k)}]', \quad q_2^{(k)} = [0 \dots 0 \; y_0^{(k)} \dots y_k^{(k)}]',$$
$$q_3^{(k)} = [0 \dots 0 \; x_k^{(k)} \dots x_0^{(k)}]', \quad q_4^{(k)} = [0 \dots 0 \; y_k^{(k)} \dots y_0^{(k)}]'. \tag{7.I.I}$$

Тогда

$$F_k = l_1' q_1^{(k-1)}, \quad l_1 = [a_{n-1} \dots a_1]', \tag{7.I.2}$$

$$G_k = l_2' q_4^{(k-1)}, \quad l_2 = [a_{-n+1} \dots a_{-1}]'. \tag{7.I.3}$$

Что же касается действий п. 3) T-алгоритма, то они выглядят следующим образом:

$$q_1^{(k)} = Z q_1^{(k-1)} + q_2^{(k-1)} s_k, \quad q_2^{(k)} = Z q_1^{(k-1)} t_k + q_2^{(k-1)}; \tag{7.I.4}$$

$$q_3^{(k)} = q_3^{(k-1)} + Z q_4^{(k-1)} s_k, \quad q_4^{(k)} = q_3^{(k-1)} t_k + Z q_4^{(k-1)}; \tag{7.I.5}$$

$$Z = \begin{bmatrix} 0 & 1 & & & \\ & 0 & 1 & & O \\ & & \ddots & \ddots & \\ O & & & 0 & 1 \\ & & & & 0 \end{bmatrix}_{(n-1)\times(n-1)} \tag{7.I.6}$$

Исходный T-алгоритм очевидно распадается на два T-алгоритма: один - связанный с (7.I.2), (7.I.4) и другой - связанный с (7.I.3), (7.I.5). Каждый из этих двух T-алгоритмов удовлетворяет условиям теоремы 6.2.4. Здесь роль матриц P_{ij} играет I и Z. Множество матриц, которые с помощью операции умножения порождаются матрицами I и Z, составляет множество целых неотрицательных степеней матрицы Z. Оно конечно, так как $Z^{n-1} = 0$, и содержит $M = n - 1$ ненулевых матриц. Согласно доказательству теоремы 6.2.4 получаем

$$L_1 = \begin{bmatrix} l_1' \\ l_1' Z \\ \dots \\ l_1' Z^{n-3} \\ l_1' Z^{n-2} \end{bmatrix} = \begin{bmatrix} a_{n-1} & a_{n-2} & \dots & \dots & a_1 \\ & a_{n-1} & a_{n-2} & \dots & a_2 \\ & & \dots & \dots & \dots \\ & O & & a_{n-1} & a_{n-2} \\ & & & & a_{n-1} \end{bmatrix}, \tag{7.I.7}$$

$$L_2 = \begin{bmatrix} l_2' \\ l_2' Z \\ \vdots \\ l_2' Z^{n-3} \\ l_2' Z^{n-2} \end{bmatrix} = \begin{bmatrix} a_{-n+1} & a_{-n+2} & \dots & \dots & a_{-1} \\ & a_{-n+1} & a_{-n+2} & \dots & a_{-2} \\ & & \dots & \dots & \dots \\ & 0 & & a_{-n+1} & a_{-n+2} \\ & & & & a_{-n+1} \end{bmatrix}. \qquad (7.I.8)$$

При этом в соотношениях $L_1 Z = Q L_1$, $L_2 Z = Q L_2$ следует взять $Q = Z$.

В целях векторизации рассматриваемых двух T -алгоритмов вводятся дополнительные $n-1$-мерные векторы

$$d_1^{(k)} = L_1 q_1^{(k)}, \quad d_2^{(k)} = L_1 q_2^{(k)},$$

$$d_3^{(k)} = L_2 q_3^{(k)}, \quad d_4^{(k)} = L_2 q_4^{(k)}. \qquad (7.I.9)$$

Естественно, полученные векторные алгоритмы объединяются в один векторный алгоритм, и при этом действия (7.I.4) и (7.I.5) можно заменить действиями п. 3) исходного алгоритма (5.2.5). Приведем полное описание векторного алгоритма для обращения тёплицевой матрицы:

$k = 0$: $\quad p_0 = q_0 = a_0^{-1}$, $\quad x_0^{(0)} = y_0^{(0)} = I$.

$$d_1^{(0)} = d_2^{(0)} = [a_1 \dots a_{n-1}]',$$

$$d_3^{(0)} = d_4^{(0)} = [a_{-1} \dots a_{-n+1}]';$$

$k = 1, \dots, n-1$: $\quad F_k = \left(d_1^{(k-1)}\right)_0, \quad G_k = \left(d_4^{(k-1)}\right)_0$,

$$s_k = -q_{k-1} F_k, \quad t_k = -p_{k-1} G_k, \qquad (7.I.10)$$

$$p_k = (I - t_k s_k)^{-1} p_{k-1}, \quad q_k = (I - s_k t_k)^{-1} q_{k-1},$$

$$x^{(k)} = \uparrow x^{(k-1)} + \downarrow y^{(k-1)} s_k, \quad y^{(k)} = \uparrow x^{(k-1)} t_k + \downarrow y^{(k-1)},$$

$$d_1^{(k)} = \uparrow d_1^{(k-1)} + d_2^{(k-1)} s_k, \quad d_2^{(k)} = \uparrow d_1^{(k-1)} t_k + d_2^{(k-1)}, \qquad (7.I.10)$$

$$d_3^{(k)} = d_3^{(k-1)} + \uparrow d_4^{(k-1)} s_k, \quad d_4^{(k)} = d_3^{(k-1)} t_k + \uparrow d_4^{(k-1)}.$$

Как видим, отсутствие k -членного суммирования для вычисления F_k и G_k компенсируется ценой дополнительного вычисления четырех векторов $d_j^{(k)}$, $1 \le j \le 4$. Параллельная форма алгоритма (7.I.10) содержит $O(n)$ ярусов.

В соответствии с замечанием 2 к теореме 6.2.2 здесь можно перейти к укороченным векторам $d_j^{(k)}$, а именно: отказаться от получения компонент $d_{j\nu}^{(k)}$ при $\nu \ge n-1+k$. После такой модификации получится векторный алгоритм, в котором выполняется $3n^2$ умножений и $3n^2$ сложений-вычитаний (в главном члене).

7.2. Ганкелевы матрицы

Рассмотрим алгоритм (5.2.I0). Его максимальная параллельная форма содержит $O(n^2)$ ярусов. Однако, будучи T -алгоритмом, он допускает векторизацию в соответствии с методом, описанным в § 6. Введем $2n-1$-мерные векторы

$$q^{(k)} = [0 \dots 0 \; y_k^{(k)} \dots y_0^{(k)}]', \qquad (7.2.I)$$

$$v^{(k)} = L q^{(k)}, \qquad (7.2.2)$$

где

$$L = \begin{bmatrix} a_{2n-2} & a_{2n-3} & \dots & a_1 & a_0 \\ & a_{2n-2} & a_{2n-3} & \dots & a_1 \\ & & \dots & \dots & \dots \\ & 0 & & a_{2n-2} & a_{2n-3} \\ & & & & a_{2n-2} \end{bmatrix}. \qquad (7.2.3)$$

Тогда

$$Q_k = v^{(k)}_k, \quad F_k = v^{(k)}_{k+1}. \tag{7.2.4}$$

Векторы $q^{(k)}$, $v^{(k)}$ удовлетворяют рекуррентным соотношениям

$$q^{(k)} = \uparrow q^{(k-1)} + q^{(k-1)}\alpha_k + q^{(k-2)}\beta_k, \tag{7.2.5}$$

$$v^{(k)} = \uparrow v^{(k-1)} + v^{(k-1)}\alpha_k + v^{(k-2)}\beta_k. \tag{7.2.6}$$

Очевидно, здесь мы действуем в духе теоремы 6.2.3; вместо (7.2.5) в полученном векторном алгоритме мы можем использовать инструкции исходного алгоритма, относящиеся к определению $y^{(k)}$. В результате векторный алгоритм для обращения ганкелевой матрицы можно сформулировать следующим образом:

$$k = 0,1: \qquad y^{(0)}_0 = y^{(1)}_1 = I, \quad y^{(1)}_0 = -a_0^{-1} a_1,$$

$$Q_0 = a_0, \quad Q_1 = a_1 y^{(1)}_0 + a_2, \quad F_0 = a_1, \quad F_1 = a_2 y^{(1)}_0 + a_3,$$

$$v^{(0)} = [a_0\ a_1\ \dots\ a_{2n-2}]', \quad v^{(1)} = \uparrow v^{(0)} + v^{(0)} y^{(1)}_0;$$

$$k = 2, \dots, n-1: \quad \alpha_k = -Q^{-1}_{k-2} Q_{k-1}, \quad \beta_k = -Q^{-1}_{k-1}(F_{k-1} + F_{k-2}\alpha_k),$$

$$\begin{aligned} y^{(k)} &= \downarrow y^{(k-1)} + \uparrow y^{(k-1)}\alpha_k + \uparrow\uparrow y^{(k-2)}\beta_k, \\ v^{(k)} &= \uparrow v^{(k-1)} + v^{(k-1)}\alpha_k + v^{(k-2)}\beta_k, \end{aligned} \tag{7.2.7}$$

$$F_k = v^{(k)}_{k+1}, \quad Q_k = v^{(k)}_k.$$

Параллельная форма алгоритма (7.2.7) содержит $O(n)$ ярусов. Здесь, как и в § 7.I, можно перейти к укороченным векторам $v^{(k)}$, а именно: отказаться от вычисления компонент $v^{(k)}_\nu$ при $\nu \le k-1$ или $2n-1 \le k \le \nu$. После указанной модификации получится векторный алгоритм, в котором выполняется $3n^2$ умножений и $3n^2$ сложений-вычитаний (в главном члене).

7.3. Тёплиц-ганкелевы матрицы

Алгоритм (5.3.9) для тёплиц-ганкелевых систем практически не

распараллеливается. Но согласно результатам, полученным в § 6, его можно преобразовать в алгоритмы, имеющий параллельную форму высотой $O(n)$. Как уже отмечалось, главное в таком преобразовании - обеспечить высокую параллельность при вычислении линейных форм.

Сначала посмотрим, как можно организовать векторное вычисление величин F_k $(2 \le k \le n-1)$. Положим

$$l \equiv [l_0 \dots l_{n-2}]' = [0, h_0 + t_{-2}, h_1 + t_{-3}, \dots, h_{n-3} + t_{-n+1}]'; \tag{7.3.I}$$

тогда $F_{k+1} = l'[y^{(k)}\ 0]'$. Рассмотрим ленточную тёплицеву матрицу

$$U = \begin{bmatrix} l_0 & l_1 & \dots & \dots & l_{n-2} & & 0 \\ & l_0 & l_1 & \dots & \dots & l_{n-2} & \\ & & \dots & \dots & \dots & \dots & \\ 0 & & & l_0 & l_1 & \dots & l_{n-2} \end{bmatrix}_{(2n-3)\times(2n-3)} \tag{7.3.2}$$

и введем дополнительные $2n-3$-мерные векторы

$$\begin{aligned} \tilde{y}^{(k)} &= U[0 \dots 0\ y^{(k)}\ 0 \dots 0]', \\ \tilde{x}^{(k)} &= U[0 \dots 0\ x^{(k)}\ 0 \dots 0]', \\ \tilde{r}^{(k)} &= U[0 \dots 0\ r^{(k)}\ 0 \dots 0]', \end{aligned} \tag{7.3.3}$$

считая, что первой компоненте векторов $y^{(k)}, x^{(k)}, r^{(k)}$ предшествует $n-2$ нулей. Ясно, что

$$F_{k+1} = y^{(k)}_{n-2}. \tag{7.3.4}$$

Далее, согласно (5.3.9), находим

$$\tilde{y}^{(k)} = \uparrow \tilde{y}^{(k-1)} + \downarrow \tilde{y}^{(k-1)}\alpha_k + \tilde{y}^{(k-2)}\beta_k - \tilde{x}^{(k-1)} F_k + \tilde{r}^{(k-1)} y^{(k-1)}_0,$$

$$\tilde{x}^{(k)} = \tilde{x}^{(k-1)} + \tilde{y}^{(k)} \varepsilon_k ,$$
$$\tilde{r}^{(k)} = \tilde{r}^{k-1} + \tilde{y}^{(k)} \tau_k . \qquad (7.3.5)$$

Теперь перейдем к векторному способу вычисления величин G_k, E_k, S_k, D_k, P_k. Для этого введем дополнительные n-мерные векторы.

$$\hat{y}^{(k)} = A\,[\, y^{(k)} \; 0 \; \ldots \; 0 \,]' ,$$
$$\hat{x}^{(k)} = A\,[\, x^{(k)} \; 0 \; \ldots \; 0 \,]' ,$$
$$\hat{r}^{(k)} = A\,[\, r^{(k)} \; 0 \; \ldots \; 0 \,]' ,$$
$$\hat{z}^{(k)} = A\,[\, \hat{z}^{(k)} \; 0 \; \ldots \; 0 \,]' .$$

Тогда

$$G_k = \hat{y}^{(k-1)}_{k+1} , \quad E_k = \hat{y}^{(k-1)}_k , \quad S_k = \hat{x}^{(k-1)}_k ,$$
$$D_k = \hat{r}^{(k-1)}_k , \quad P_k = \hat{z}^{(k-1)}_k . \qquad (7.3.6)$$

Получим сначала рекуррентную формулу для $\hat{y}^{(k)}$. Это можно сделать, опираясь на соотношение (5.3.3). Пусть матрица Z_k имеет вид (5.3.2). Тогда

$$A \begin{bmatrix} Z_k \begin{bmatrix} y^{(k-1)} \\ 0 \end{bmatrix} \\ 0 \\ \ldots \\ 0 \end{bmatrix} = A Z_{n-1} \begin{bmatrix} y^{(k-1)} \\ 0 \\ \ldots \\ 0 \end{bmatrix} + a^{(k+1)} ,$$

где

$$a^{(k+1)} = [\, a_{0,k+1} \; \ldots \; a_{n-1,k+1} \,]' .$$

В силу (5.3.3), (5.3.4) имеем

$$A Z_{n-1} \begin{bmatrix} y^{(k-1)} \\ 0 \\ \ldots \\ 0 \end{bmatrix} = Z_{n-1} \hat{y}^{(k-1)} - u^{(n-1)} y_0^{(k-1)} +$$
$$+ [\, F_k \; 0 \; \ldots \; 0 \; G_k \,]'$$

Следовательно,

$$\hat{y}^{(k)} = a^{(k+1)} + Z_{n-1} \hat{y}^{(k-1)} - u^{(n-1)} y_0^{(k-1)} +$$
$$+ [\, F_k \; 0 \ldots 0 \; G_k \,]' + \hat{y}^{(k-1)} \alpha_k + \hat{y}^{(k-2)} \beta_k - \hat{x}^{(k-1)} F_k + \hat{r}^{(k-1)} y_0^{(k-1)} \qquad (7.3.7)$$

Коэффициенты α_k, β_k определяются согласно предписаниям алгоритма (5.3.9); $F_k = \tilde{y}^{(k-1)}_{n-1}$. Кроме того, имеем

$$\hat{x}^{(k)} = \hat{x}^{(k-1)} + \hat{y}^{(k)} \varepsilon_k ,$$
$$\hat{r}^{(k)} = \hat{r}^{(k-1)} + \hat{y}^{(k)} \tau_k , \qquad (7.3.8)$$
$$\hat{z}^{(k)} = \hat{z}^{(k-1)} + \hat{y}^{(k)} \omega_k .$$

Соотношения (7.3.4)-(7.3.8) дают описание искомого векторного алгоритма, имеющего параллельную форму с числом ярусов $O(n)$, точнее, той его части, которая содержит новый способ вычисления линейных форм из T-алгоритма (5.3.9). Обратим внимание также на возможность перехода к укороченным векторам.

7.4. Матрицы малого тёплицева ранга

Проведем здесь векторизацию алгоритма (5.4.I2). Чтобы преобразовать способ вычисления скалярных величин (блоков) μ_{sk} $(1 \le s \le t)$, введем n-мерные векторы

$$\hat{y}^{(k)} = A \begin{bmatrix} y^{(k)} \\ 0 \end{bmatrix} , \quad \hat{w}_s^{(k)} = A \begin{bmatrix} w_s^{(k)} \\ 0 \end{bmatrix} , \quad s = 1, \ldots, t . \qquad (7.4.I)$$

Тогда

$$\mu_{sk} = \alpha_{sk} - \left(\hat w_s^{(k)}\right)_k, \quad s = 1, \dots, t. \qquad (7.4.2)$$

Учитывая, что матрица A задается своим тёплицевым разложением (4.I.3), находим

$$A\begin{bmatrix}0\\ y^{(k-1)}\\ 0\end{bmatrix} = \begin{bmatrix}0 & 0 & \dots & 0\\ 0 & & & \\ \dots & & A_{n-2} & \\ 0 & & & \end{bmatrix}\begin{bmatrix}0\\ y^{(k-1)}\\ 0\end{bmatrix} +$$

$$+\begin{bmatrix}\alpha_{10} & \dots & \alpha_{t0}\\ \alpha_{11} & \dots & \alpha_{t1}\\ \dots & \dots & \dots\\ \alpha_{1,n-1} & \dots & \alpha_{t,n-1}\end{bmatrix}\begin{bmatrix}\beta_{10} & \beta_{11} & \dots & \beta_{1,n-1}\\ \dots & \dots & \dots & \dots\\ \beta_{t0} & \beta_{t1} & \dots & \beta_{t,n-1}\end{bmatrix}\begin{bmatrix}0\\ y^{(k-1)}\\ 0\end{bmatrix} =$$

$$= \downarrow\hat y^{(k-1)} + \begin{bmatrix}\alpha_{10} & \dots & \alpha_{t0}\\ \alpha_{11} & \dots & \alpha_{t1}\\ \dots & \dots & \dots\\ \alpha_{1,n-1} & \dots & \alpha_{t,n-1}\end{bmatrix}\begin{bmatrix}\nu_{1k}\\ \dots\\ \nu_{tk}\end{bmatrix} = \downarrow\hat y^{(k-1)} + \alpha\begin{bmatrix}\nu_{1k}\\ \dots\\ \nu_{tk}\end{bmatrix}. \qquad (7.4.3)$$

Вследствие этого в соответствии с предписаниями алгоритма (5.4.I2) получаем

$$\hat y^{(k)} = \downarrow\hat y^{(k-1)} + \alpha\begin{bmatrix}\nu_{1k}\\ \dots\\ \nu_{tk}\end{bmatrix} - \sum_{s=1}^{t} \hat w_s^{(k-1)}\nu_{sk}, \qquad (7.4.4)$$

$$\hat w_s^{(k)} = \hat w_s^{(k-1)} + \hat y^{(k)}\zeta_{sk}, \quad s = 1, \dots, t.$$

Далее, чтобы преобразовать способ вычисления скалярных величин (блоков) ν_{sk} $(1 \le s \le t)$, используем конструкцию из доказательства

теоремы 6.2.4. Здесь потребуется ввести $t(t+1)$ дополнительных векторов размерности $n-1$:

$$\tilde y_s^{(k)} = B_s\left[0 \dots 0\; y_k^{(k)} \dots y_0^{(k)}\right]', \quad 1 \le s \le t;$$

$$\tilde w_{sr}^{(k)} = B_s\left[0 \dots 0\; (w_r^{(k)})_k \dots (w_r^{(k)})_0\right]', \quad 1 \le s, r \le t; \qquad (7.4.5)$$

$$B_s' = \begin{bmatrix}\beta_{s,n-1} & \beta_{s,n-2} & \dots & \beta_{s,2} & \beta_{s,1}\\ & \beta_{s,n-1} & \beta_{s,n-2} & \dots & \beta_{s,2}\\ & & \dots & \dots & \dots\\ & 0 & & \beta_{s,n-1} & \beta_{s,n-2}\\ & & & & \beta_{s,n-1}\end{bmatrix}, \qquad (7.4.6)$$

$1 \le s \le t.$

Очевидно, $\nu_{sk} = \left(\tilde y_s^{(k-1)}\right)_0 (1 \le s \le t)$.

Приведем ниже полное описание векторного алгоритма для решения системы (5.I.I) с (блочной) матрицей A, заданной тёплицевым разложением (4.I.3):

$$k = 0: \quad a_{i0} = \sum_{s=1}^{t} \alpha_{si}\beta_{s0}, \quad 0 \le i \le n-1$$

$$y_0^{(0)} = I, \quad q_0 = a_{00}^{-1}; \quad z_0^{(0)} = q_0 b_0,$$

$$(w_s^{(0)})_0 = q_0\alpha_{s0}, \quad 1 \le s \le t,$$

$$\tilde y_s^{(0)} = \left[\beta_{s1} \dots \beta_{s,n-1}\right]', \quad 1 \le s \le t,$$

$$\tilde w_{sr}^{(0)} = \left[\beta_{s1} \dots \beta_{s,n-1}\right]'(w_r^{(0)})_0, \quad 1 \le s, r \le t, \qquad (7.4.7)$$

$$\hat y^{(0)} = \left[a_{00} \dots a_{n-1,0}\right]',$$

$$\hat w_s^{(0)} = \left[a_{00} \dots a_{n-1,0}\right]'(w_s^{(0)})_0, \quad 1 \le s \le t,$$

$$\hat z^{(0)} = \left[a_{00} \dots a_{n-1,0}\right] z_0^{(0)};$$

8957

20-2

$$k = 1, \ldots, n-1: \qquad \mu_{sk} = d_{sk} - (\hat{w}_s^{(k)})_k, \quad 1 \le s \le t,$$

$$\nu_{sk} = (\tilde{y}_s^{(k)})_0, \quad 1 \le s \le t,$$

$$\varepsilon_k = I + \left(\sum_{s=1}^{t} \mu_{sk} \nu_{sk} \right) q_{k-1}, \quad q_k = q_{k-1} \varepsilon_k^{-1},$$

$$\omega_k = q_k \left(b_k - (\hat{z}^{(k-1)})_k \right),$$

$$\zeta_{sk} = q_k \mu_{sk}, \quad 1 \le s \le t,$$

$$y^{(k)} = t\, y^{(k-1)} - \sum_{s=1}^{t} t\, w_s^{(k-1)} \nu_{sk},$$

$$w_s^{(k)} = t\, w_s^{(k-1)} + y^{(k)} \zeta_{sk}, \quad 1 \le s \le t,$$

$$z^{(k)} = t\, z^{(k-1)} + y^{(k)} \omega_k, \tag{7.4.7}$$

$$\hat{y}^{(k)} = t\, \hat{y}^{(k-1)} + d \begin{bmatrix} \nu_{1k} \\ \cdots \\ \nu_{tk} \end{bmatrix} - \sum_{s=1}^{t} \hat{w}_s^{(k-1)} \nu_{sk},$$

$$\hat{w}_s^{(k)} = \hat{w}_s^{(k-1)} + \hat{y}^{(k)} \zeta_{sk}, \quad 1 \le s \le t,$$

$$\hat{z}^{(k)} = \hat{z}^{(k-1)} + \hat{y}^{(k)} \omega_k,$$

$$\tilde{y}_s^{(k)} = t\, \tilde{y}_s^{(k-1)} - \sum_{r=1}^{t} \tilde{w}_{sr}^{(k-1)} \nu_{rk}, \quad 1 \le s \le t,$$

$$\tilde{w}_{sr}^{(k)} = \tilde{w}_{sr}^{(k-1)} + \hat{y}_s^{(k)} \zeta_{rk}, \quad 1 \le s, r \le t.$$

Полученный алгоритм обладает параллельной формой с числом ярусов $O(tn)$. Число дополнительно вычисляемых векторов равно t^2+2t+3. Заметим, что здесь, как и в других построенных выше векторных алгоритмах, имеется возможность перехода к укороченным векторам.

7.5. Матрицы типа Коши

Рассмотрим алгоритм (5.5.I9) для решения линейной алгебраической системы (5.I.I) с блочной матрицей типа Коши, имеющей скалярные

блоки x_i, y_i. Очевидно, можно считать x_i, y_i просто числами. Алгоритм (5.5.I9) является T-алгоритмом и поэтому может быть векторизован в соответствии с построениями, выполненными в § 6. Для этого введем дополнительные величины

$$f_{lm}^{(k)} = \varphi_{lm} - \sum_{j=0}^{k} a_{mj} u_{lj}^{(k)}, \quad g_{lm}^{(k)} = \psi_{lm} - \sum_{j=0}^{k} v_{lj}^{(k)} a_{jm},$$

$$1 \le l \le r, \quad k+1 \le m \le n-1; \tag{7.5.I}$$

$$p_m^{(k)} = \sum_{j=0}^{k} a_{mj} c_j^{(k)}, \quad q_m^{(k)} = \sum_{j=1}^{k} s_j^{(k)} a_{jm},$$

$$k+1 \le m \le n-1.$$

Тогда

$$f_{lm}^{(k)} = f_{lm}^{(k-1)} - d_{lk} p_m^{(k)}, \quad g_{lm}^{(k)} = g_{lm}^{(k-1)} - q_m^{(k)} \beta_{lk}, \tag{7.5.2}$$

$$1 \le l \le r, \quad k+1 \le m \le n-1,$$

и вместе с тем имеем

$$d_{lk} = f_{lk}^{(k-1)}, \quad \beta_{lk} = g_{lk}^{(k-1)}, \quad 1 \le l \le r. \tag{7.5.3}$$

Остается найти способ векторизованного вычисления величин $p_m^{(k)}$ и $g_m^{(k)}$. Согласно (5.5.6), (5.5.9) запишем

$$p_m^{(k)} = \sum_{j=0}^{k} a_{mj} \frac{1}{x_k - y_j} \sum_{l=1}^{r} u_{lj}^{(k)} v_{lk}^{(k)} =$$

$$= \sum_{j=0}^{k-1} a_{mj} \frac{1}{x_k - y_j} \sum_{l=1}^{r} u_{lj}^{(k-1)} v_{lk}^{(k)} + (x_k - y_k) \sum_{j=0}^{k} a_{mj} c_j^{(k)} \frac{1}{x_k - y_j}.$$

Положим $\hat{a}_{mj} = a_{mj}(x_m - y_j)$. Учитывая равенство

$$\frac{1}{(x_m - y_j)(x_k - y_j)} = \frac{1}{x_m - x_k} \left(\frac{1}{x_k - y_j} - \frac{1}{x_m - y_j} \right), \tag{7.5.4}$$

находим

$$p_m^{(k)} = \frac{1}{x_m-x_k}\left(\sum_{j=0}^{k-1}\hat a_{mj}\frac{1}{x_k-y_j}\sum_{l=1}^{r}u_{lj}^{(k-1)}v_{lk}^{(k)}\; -\right.$$

$$\left.-\sum_{j=0}^{k}\hat a_{mj}\frac{1}{x_m-y_j}\sum_{l=1}^{r}u_{lj}^{(k-1)}v_{lk}^{(k)}\right)+ \tag{7.5.5}$$

$$+\frac{x_k-y_k}{x_m-x_k}\left(\sum_{j=0}^{k}\hat a_{mj}c_j^{(k)}\frac{1}{x_k-y_j}-\sum_{j=0}^{k}\hat a_{mj}c_j^{(k)}\frac{1}{x_m-y_j}\right).$$

Последняя сумма (из четырех) есть в точности $p_m^{(k)}$. Вторая сумма в первых скобках имеет вид

$$\sum_{l=1}^{r}\sum_{j=0}^{k}a_{mj}u_{lj}^{(k-1)}v_{lk}^{(k)} = \sum_{l=1}^{r}\left(\varphi_{lm}-f_{lm}^{(k-1)}\right)v_{lk}^{(k)}. \tag{7.5.6}$$

Кроме того, с учетом (5.5.9) имеем

$$\sum_{j=0}^{k-1}\hat a_{mj}\frac{1}{x_k-y_j}\sum_{l=1}^{r}u_{lj}^{(k-1)}v_{lk}^{(k)}\; -$$

$$-\sum_{j=0}^{k}\hat a_{mj}c_j^{(k)}\frac{x_k-y_k}{x_k-y_j} = \sum_{j=0}^{k}\hat a_{mj}c_j^{(k)}. \tag{7.5.7}$$

Таким образом, вследствие (7.5.5) получаем

$$(x_m-x_k)\left(1+\frac{x_k-y_k}{x_m-x_k}\right)p_m^{(k)} = (x_k-y_k)\,p_m^{(k)} =$$

$$=\sum_{l=1}^{r}f_{lm}^{(k-1)}v_{lk}^{(k)}-\sum_{l=1}^{r}\varphi_{lm}v_{lk}^{(k)}+\sum_{j=0}^{k}\hat a_{mj}c_j^{(k)} \tag{7.5.8}$$

Теперь примем во внимание равенство

$$v_{lk}^{(k)} = \left[v_{l0}^{(k)}\ \dots\ v_{lk}^{(k)}\right]\begin{bmatrix}0\\ \vdots\\ 0\\ 1\end{bmatrix} = \left[v_{l0}^{(k)}\ \dots\ v_{lk}^{(k)}\right]A_k\begin{bmatrix}c_0^{(k)}\\ \vdots\\ c_k^{(k)}\end{bmatrix} =$$

$$=\sum_{j=0}^{k}\psi_{lj}c_j^{(k)}.$$

Оно позволяет записать

$$\sum_{j=0}^{k}\hat a_{mj}c_j^{(k)} = \sum_{j=0}^{k}\left(\sum_{l=1}^{r}\varphi_{lm}\psi_{lj}\right)c_j^{(k)} =$$

$$=\sum_{l=1}^{r}\varphi_{lm}\sum_{j=0}^{k}\psi_{lj}c_j^{(k)} = \sum_{l=1}^{r}\varphi_{lm}v_{lk}^{(k)}$$

Следовательно, согласно (7.5.8) находим

$$p_m^{(k)} = \frac{1}{x_m-y_k}\sum_{l=1}^{r}f_{lm}^{(k-1)}v_{lk}^{(k)} \tag{7.5.9}$$

$$k+1\le m\le n-1.$$

Аналогично,

$$q_m^{(k)} = \frac{1}{x_k-y_m}\sum_{l=1}^{r}u_{lk}^{(k)}g_{lm}^{(k-1)}, \tag{7.5.10}$$

$$k+1\le m\le n-1.$$

Искомый векторный алгоритм – точнее, векторный способ вычисления величин α_{lk}, β_{lk} – определяется формулами (7.5.2), (7.5.3), (7.5.9), (7.5.10). Чтобы получить еще и величины γ_k, вычисляемые в (5.5.19), введем векторы

$$\hat z^{(k)} = A\begin{bmatrix}z^{(k)}\\ 0\end{bmatrix}. \tag{7.5.11}$$

Тогда

$$\gamma_k = b_k - \hat{z}_k^{(k-1)}, \qquad (7.5.12)$$

а векторы $\hat{z}^{(k)}$ вычисляются рекуррентно следующим способом:

$$\hat{z}^{(k)} = \hat{z}^{(k-1)} + \left[0 \dots 0 \; I \; \rho_{k+1}^{(k)} \dots \rho_{n-1}^{(k)}\right]' \gamma_k . \qquad (7.5.13)$$

Построенный векторный алгоритм имеет параллельную форму высотой $O(n)$.

§ 8. Дихотомия в векторных алгоритмах

8.I. Быстрые алгоритмы для тёплицевых систем

Рассмотрим систему (5.I.I), считая, что матрица $A=[a_{i-j}]_{ij=0}^{n-1}$ блочно-тёплицева, блочного порядка n, с блоками порядка p. Будем опираться на то, что процесс решения системы может быть разделен на два этапа: I) вычисление тёплицева разложения матрицы A^{-1}; 2) умножения A^{-1} на вектор правой части с использованием ее тёплицева разложения. Этап 2) реализуется ценой $O(n\log_2 n)$ операций, так как сводится к нескольким умножениям на вектор тёплицевых матриц. Поэтому если мы хотим ускорить процесс в целом, то это следует сделать в отношении этапа I).

Предположим, что матрица A невырожденная вместе со всеми ведущими подматрицами $A_k=[a_{i-j}]_{ij=0}^{k}$. Начнем с того, что возьмем (как основу для наших построений) векторный алгоритм (7.I.I0) и получим для него полиномиальное представление, т.е. заменим операции с векторами на операции с многочленами. Блочные многочлены от переменной z нулевого шага определим таким образом:

$$x^{(0)}(z) = y^{(0)}(z) = I,$$

$$\beta_i^{(0)}(z) \equiv \sum_{j=0}^{n-2} \beta_{ij}^{(0)} z^j = \sum_{j=0}^{n-2} d_{i,n-2-j}^{(0)} z^j, \quad i=1,2,3,4. \qquad (8.I.I)$$

Далее, многочлены k-го шага введем с помощью следующих рекуррентных соотношений

$$\left[x^{(k)}(z)\, y^{(k)}(z)\right] = \left[x^{(k-1)}(z)\, y^{(k-1)}(z)\right] \begin{bmatrix} I & t_k \\ s_k z & z \end{bmatrix}, \qquad (8.I.2)$$

$$\left[\beta_1^{(k)}(z)\, \beta_2^{(k)}(z)\right] = \left[\beta_1^{(k-1)}(z)\, \beta_2^{(k-1)}(z)\right] \begin{bmatrix} z & t_k z \\ s_k & I \end{bmatrix}, \qquad (8.I.3)$$

$$\left[\beta_3^{(k)}(z)\, \beta_4^{(k)}(z)\right] = \left[\beta_3^{(k-1)}(z)\, \beta_4^{(k-1)}(z)\right] \begin{bmatrix} I & t_k \\ s_k z & z \end{bmatrix}. \qquad (8.I.4)$$

Сравнивая эти формулы с (7.I.I0), приходим к выводу: коэффициент при z^j в многочленах $x^{(k)}(z)$ и $y^{(k)}(z)$ совпадает с j-й компонентой соответственно векторов $x^{(k)}$ и $y^{(k)}$; коэффициент при z^j в многочлене $\beta_i^{(k)}(z)$ совпадает с $n-2-j$-й компонентой вектора $d_i^{(k)}$ $(i=1,2,3,4)$. Заметим также, что алгоритм, реализующий непосредственно формулы (8.I.I)-(8.I.4), содержит некоторые избыточные вычисления по сравнению с (7.I.I0). Тем не менее, чтобы построить быстрый алгоритм, мы будем опираться именно на формулы (8.I.I)-(8.I.4). В интересах сокращения записи положим

$$M^k = \left\{\beta_i^{(k)}(z),\ i=1,2,3,4\right\}. \qquad (8.I.5)$$

Введем в рассмотрение матрицы

$$R^{kl}(z) = \left[r_{ij}^{kl}(z)\right]_{ij=0}^{1} = \begin{bmatrix} I & t_{k+1} \\ s_{k+1} z & z \end{bmatrix} \begin{bmatrix} I & t_{k+2} \\ s_{k+2} z & z \end{bmatrix} \cdots \begin{bmatrix} I & t_l \\ s_l z & z \end{bmatrix}, \qquad (8.I.6)$$

$$U^{kl}(z) = \left[u_{ij}^{kl}(z)\right]_{ij=0}^{1} = \begin{bmatrix} z & t_{k+1} z \\ s_{k+1} & I \end{bmatrix} \begin{bmatrix} z & t_{k+2} z \\ s_{k+2} & I \end{bmatrix} \cdots \begin{bmatrix} z & t_l z \\ s_l & I \end{bmatrix}, \qquad (8.I.7)$$