Теорія оптимальних рішень

УДК 519.6

В.В. ПОЛЯНКО

ОПТИМІЗАЦІЯ СТРУКТУРИ РОЗРІДЖЕНОЇ МАТРИЦІ ДЛЯ ПОБУДОВИ ЕФЕКТИВНОГО ОБЧИСЛЮВАЛЬНОГО АЛГОРИТМУ

Розглядаються проблеми оптимізації структури розрідженої матриці для побудови ефективного обчислювального алгоритму знаходження розв'язку системи лінійних алгебраїчних рівнянь. Запропоновано кілька підходів до модифікації матриці та їх застосування в ітераційному методі розв'язування системи.

Вступ. Значна кількість прикладних задач включає у себе як складову частину знаходження розв'язку системи лінійних алгебраїчних рівнянь (СЛАР) з розрідженими матрицями. За останні роки у світі розроблено чимало алгоритмів розв'язування подібних СЛАР, у тому числі й паралельних [1]. Серед розроблених методів немає універсального, який би ефективно розв'язував системи з матрицями будь-якого типу розрідженості, але є такі, що призначені для матриць певної структури [2]. Тому одним з важливих завдань при розв'язуванні СЛАР є попередня оптимізація структури розрідженої матриці, що передбачає можливість ефективного застосування паралельних алгоритмів знаходження розв'язку системи.

Залежність часу розв'язування системи від характеру розрідженості матриці особливо відчутна при реалізації ітераційних методів. Адже, від способу побудови передобумовлювача залежить не лише час знаходження розв'язку і кількість виконаних ітерацій, але й збіжність методу взагалі. Тому основна мета роботи – розгляд методів оптимізації структури матриці для передобумовлювача в ітераційному методі розв'язування СЛАР.

Постановка задачі. Розглянемо систему лінійних алгебраїчних рівнянь:

$$Ax = b, \qquad (1)$$

де A – симетрична розріджена додатно визначена матриця, x –вектор розв'язку

системи, b – вектор правих частин.

Нехай для розв'язування СЛАР (1) використовується метод спряжених градієнтів [3]:

$$\begin{gathered} Bx_{k+1} = \alpha_{k+1}(B - \tau_{k+1}A)x_k + (1 - \alpha_{k+1})Bx_{k-1} + \alpha_{k+1}\tau_{k+1}f, \\ k = 1, 2,\ldots, \\ Bx_1 = (B - \tau_1 A)x_0 + \tau_1 f,\ x_0 \in H, \end{gathered} \tag{2}$$

де ітераційні параметри α_{k+1} і τ_{k+1} обчислюються за формулами

$$\tau_{k+1} = \frac{(\omega_k, r_k)}{(A\omega_k, \omega_k)}, k = 0,1,..., \alpha_{k+1} = \left(1 - \frac{\tau_{k+1}}{\tau_k}\frac{(\omega_k, r_k)}{(\omega_{k-1}, r_{k-1})}\frac{1}{\alpha_k}\right)^{-1}, k = 1,2,..., \alpha_1 = 1, \tag{3}$$

а сам обчислювальний алгоритм має вигляд:

1) за заданим x_0 обчислюється нев'язка $r_0 = A\,x_0 - f$;
2) розв'язується рівняння поправок $B\omega_0 = x_0$;
3) з формули (3) обчислюється параметр τ_1;
4) наближення x_1 знаходься з формули $x_{k+1} = x_k - \tau_{k+1}\omega_k$;

далі для $k = 1, 2,\ldots$, послідовно виконуються наступні дії:

5) обчислюється нев'язка $r_k = A\,x_k - f$;
6) розв'язується рівняння поправок

$$B\omega_k = x_k; \tag{4}$$

7) за формулами (3) обчислюються параметри α_{k+1} і τ_{k+1};
8) знаходиться нове наближення $x_{k+1} = \alpha_{k+1}x_k + (1 - \alpha_{k+1})x_{k-1} - \alpha_{k+1}\tau_{k+1}\omega_k$.

Завдання побудови ефективного обчислювального алгоритму полягає у знаходженні оптимального передобумовлювача B. Легко бачити, що, використовуючи наведену схему, ми перейшли від задачі розв'язування системи (1) до задачі розв'язування системи (4). Отже, саме у виборі матриці B слід шукати способи прискорення знаходження розв'язку вихідної СЛАР.

При пошуку виду матриці B вирізняють дві головні вимоги. По-перше, передобумовлювач має бути якомога «ближчим» до вихідної матриці. Чим менше відмінностей між матрицями передобумовлювача і системи, тим меншу кількість ітерацій необхідно буде виконати під час роботи ітераційного методу. А по-друге, його структура має бути досить простою для проведення обчислень. Ідеальним варіантом з точки зору першого положення є матриця передобумовлювача, ідентична до матриці системи. У цьому випадку метод досягне збіжності за одну ітерацію. Проте обчислювальні витрати дорівнюватимуть витратам прямого методу розв'язування СЛАР. Для другого положення найкращим варіантом матриці передобумовлювача є діагональна матриця. Обчислювальні витрати при її обробці мінімальні, але у цьому разі кількість ітерацій значно зростає. Тому оптимальну структуру матриці B слід шукати десь посередині.

Методи неповної факторизації. Одним із класів методів, які дозволяють ефективно поєднати наведені дві вимоги з якомога меншим впливом негативних факторів є схеми, побудовані на основі неповної *LU*-факторизації (НФ).

У методі повної *LU*-факторизації матриця системи подається у вигляді

$$A = L_A U_A, \tag{5}$$

де L_A і U_A нижня і верхня трикутні матриці відповідно.

Подібне представлення передбачає розв'язування СЛАР вигляду $Ax = b$ шляхом виконання прямого ходу для нижньої трикутної системи $L_A y = b$, а потім зворотного ходу для верхньої трикутної системи $U_A x = y$. Проте такий алгоритм призводить до заповнення портрету матриці системи. Тобто, у частинах L_A і U_A з'являються ненульові елементи на тих позиціях, де у вихідній матриці A знаходилися нулі. Як наслідок, зростає кількість арифметичних операцій, а також збільшується об'єм пам'яті, необхідний для зберігання матриць.

Ідея зменшення кількості нових ненульових елементів лежить в основі методів неповної *LU*-факторизації [4]. Метод, при якому заповнення не відбувається взагалі, отримав назву *ILU*-факторизації або факторизації з нульовим заповненням *ILU*(0). Розглянемо його детальніше.

Замість задачі знаходження факторизації (5) сформулюємо іншу задачу. Для заданої матриці A вимагатимемо подати її у вигляді $A = LU + E$, з виконанням наступних вимог:

1) матриці L і U є нижньою та верхньою трикутною відповідно;
2) множини елементів портретів L і U – підмножини портрету A;
3) для будь-якого ненульового елемента a_{ij} матриці A виконується рівність: $[LU]_{ij} = [a_{ij}]$;
4) портрети матриць A та E не мають спільних елементів.

Метод неповної факторизації з нульовим заповненням полягає у виконанні виключення Гаусса з відкиданням усіх елементів у L та U, які знаходяться поза портретом матриці A. Така схема досить проста у реалізації і вимагає меншу кількість арифметичних операцій для виконання, ніж повна факторизація. Проте, у випадку, коли матриця похибки E достатньо велика, кількість ітерацій сильно зростає, а для деяких задач збіжності взагалі може не бути. Тому різноманітні модифікації неповної факторизації направлені на зменшення E.

Одним з таких методів є неповна факторизація з рівнями заповнення ILU(p) [5]. Ідея методу наступна. Спочатку рівні заповнення всіх ненульових елементів у A дорівнюють 0. Коли елемент $f_{i,j}$ на позиції (i, j) модифікується за формулою $f_{ij} = - l_{ik} u_{kj}$ його рівень заповнення обчислюється, як

$$\text{рівень } (f_{ij}) = \text{рівень } (l_{ik}) + \text{рівень } (u_{kj}) + 1.$$

Таке систематичне означення дозволяє отримати зручну стратегію для відкидання елементів: ігноруються лише елементи, рівень яких перевищує p. Зазна-

чимо, що при $p = 0$ алгоритм неповної факторизації з рівнями заповнення ідентичний алгоритму неповної факторизації без заповнення.

Алгоритм неповної факторизації з рівнями заповнення реалізує схему символьної оптимізації заповнення матриці. Проте у ній ніяк не враховуються числові значення ненульових елементів. Отже, може статися, що зберігатимуться елементи з малими абсолютними значеннями тоді, як деякі елементи з більшими абсолютними значеннями будуть відкинуті. Оскільки більші числові значення у загальному випадку більше впливають на обчислення, ніж малі, то це може призвести до відчутнішої втрати точності неповної факторизації і, як наслідок, до зростання кількості ітерацій для досягнення збіжності при знаходженні розв'язку системи. Виходом з цієї ситуації є використання методів розвинення матриці, які при відкиданні елементів ураховують їх абсолютні значення. Одним з найпоширеніших є алгоритм неповної факторизації з порогами ILUT(τ, p).

Суть алгоритму полягає у тому, що при обробці кожного рядка елементи, значення яких не досягають певного числового порогу τ, відкидаються, а з решти вибираються для подальших обчислень лише p найбільших. Порогове значення τ відсіює ненульові елементи, які менше впливають на обчислення, а обмеження у кількості p дозволяє тримати під контролем заповнення матриці.

Результати обчислень. Алгоритм оптимізації структури передобумовлювача на основі неповної факторизації було реалізовано програмно. Тестування навіть невеликих задач показало, що ітераційний метод дуже чутливий до алгоритму впорядкування ненульових елементів матриці. Але, якщо для прямого методу найкращим був алгоритм мінімальної степені [6], то для ітераційного краще застосовувати ті, які зменшують розкид елементів щодо діагоналі. Враховуючи, що завдяки методу неповної факторизації початково нульові елементи, які мали б змінити значення, не враховуються, тобто застосування методу попереднього впорядкування не змінює кількості елементів, які братимуть участь у розв'язуванні СЛАР, а лише впливає на їх розташування. Тому більш важливим є не загальна кількість ненульових елементів, які могли би з'явитися, а кількість таких елементів, породжених одним ненульовим. Адже чим довший такий ланцюжок нових елементів, тим більше відхилення результату неповної факторизації від повної. З цієї точки зору, для точності наближення краще мати більше елементів, кожен з яких провокує появу невеликої кількості ненульових елементів (у випадку повної факторизації), ніж меншу кількість елементів, але таких що провокують появу більших кількостей нових елементів. Наприклад, якщо порівнювати метод Катхіла–Макі та алгоритм мінімальної степені, то для даної задачі краще застосувати метод перший, бо він зменшує профіль матриці, а останній збільшує розкид ненульових елементів щодо діагоналі.

Підібравши оптимальну структуру, можна отримати економію часу в порівнянні з прямим методом. Також, однією з переваг неповної факторизації є те, що матриця системи має меншу щільність, ніж у прямому методі, що призводить до зменшення кількості арифметичних операцій і, як наслідок, загального часу розв'язування СЛАР (табл. 1).

ТАБЛИЦЯ 1. Порівняння часів роботи ітераційного (НФ) та прямого методів

Порядок матриці	Ширина стрічки	Щільність матриці (%)		Ітераційний метод		Прямий (сек.)
		початкова	після НФ	ітерацій	час(сек.)	
10000	100	2.0098	0.088804	54	**1**	**1**
20000	100	1.0099	0.044551	83	**7**	**2**
40000	100	0.5062	0.022313	128	**49**	**5**
80000	100	0.253412	0.011166	239	**352**	**11**
10000	200	3.9499	0.088504	63	**1**	**2**
20000	200	1.99495	0.044551	95	**6**	**6**
40000	200	1.00248	0.02235	148	**35**	**12**
80000	200	0.502487	0.011194	234	**196**	**24**
10000	400	7.70995	0.087454	22	**0**	**10**
20000	400	3.93498	0.044326	38	**3**	**21**
40000	400	1.98749	0.022313	65	**13**	**43**
80000	400	0.998744	0.011119	108	**63**	**85**

Блочно-діагональний підхід. Ще одним способом побудови передобумовлювача є модифікації існуючих методів впорядкування ненульових елементів матриці. Наприклад, відомо, що метод паралельних перерізів [6] зводить портрет розрідженої матриці до блочно-діагонального (БД) вигляду з обрамленням. Оскільки B відрізняється від A, можемо відкинути обрамлення і отримати «легку» для обчислень блочно-діагональну матрицю. Розглянемо схему детальніше.

Метод паралельних перерізів впорядкування ненульових елементів матриці полягає у наступному. В графі матриці вибирається псевдопериферійна вершина, з коренем в якій будується структура рівнів. У залежності від довжини та ширини структури знаходиться оптимальна кількість блоків, на які структура розрізається паралельними роздільниками. Після цього поблочно, починаючи від кореня, відбувається перенумерація вершин. Усі вершини, що потрапили до роздільників, нумеруються в останню чергу, складаючи, таким чином, останній блок. У результаті цієї процедури усі вершини, що зв'язують блоки, потрапля-

ють до останнього блоку. Саме вони складають обрамлення матриці. Решта ненульових елементів розташовуються у блоках на діагоналі.

Легко бачити, що, нехтуючи обрамленням, факторизація отриманої матриці розпадається на ряд дрібних задач факторизації окремих блоків, що не пов'язані між собою. Таким чином, досягається кілька переваг. По-перше, зменшуються вимоги до оперативної пам'яті, оскільки при перетворенні матриці ми оперуємо не з цілими рядками, а лише з їхніми частинами, що знаходяться у межах блоків. І економія тим значніша, чим більша кількість блоків, адже тоді вони мають менші розміри. По-друге, з'являється можливість одночасних незалежних обчислень у різних блоках. Це особливо важливо для реалізації алгоритму з паралельною організацією обчислень, адже у такому випадку кількість міжпроцесних обмінів зводиться до мінімуму.

Ще однією важливою особливістю розглядуваного підходу є спричинене способом перевпорядкування, збереження додатної визначеності матриці. Адже ця властивість є необхідною умовою збіжності ітераційного методу. З цієї точки зору блочно-діагональний підхід має суттєву перевагу перед неповною факторизацією, де додатня визначеність не гарантується.

Результати обчислень. Алгоритм оптимізації структури матриці з використанням методу паралельних перерізів з наступним відкиданням обрамлення було реалізовано програмно. Програма виконала ряд тестів з метою порівняння часів розв'язування СЛАР прямим методом та ітераційним методом з передобумовлювачем у вигляді матриці блочно-діагональної структури (табл. 2).

ТАБЛИЦЯ 2. Порівняння часів роботи ітераційного (БД) та прямого методів

Порядок 20 000, ширина стрічки 400							
	Число блоків:	1	2	4	8	16	32
Ітераційний метод	Час роботи (с.)	12	5	4	3	2	3
	Прогноз n	12	2.5	1	0.38	0.13	0.09
	Число ітерацій	2	20	44	58	91	111
Прямий метод	Час роботи (с.)	12	6	6	7	19	70
	Прогноз n	12	3	1.5	0.86	0.20	0.63

Порядок 100 000, ширина стрічки 400							
	Число блоків:	1	2	4	8	16	31
Ітераційний метод	Час роботи (с.)	73	97	115	132	159	202
	Прогноз n	73	48.5	28.8	16.5	9.94	6.52
	Число ітерацій	2	69	127	183	255	355
Прямий метод	Час роботи (с.)	73	120	270	748	2786	9340
	Прогноз n	73	60	67.5	93.5	174.4	301.3

Виявилося, що розбиття матриці вже на невелику кількість блоків для деяких матриць дає перевагу в часі в 2–3 рази. Якщо врахувати, що обчислення в межах окремих блоків здійснюються незалежно, то при паралельній організації алгоритму можна досягти прискорення ще в кілька разів. Прогнозоване значення для паралельного випадку, де кількість процесів відповідає числу блоків подано у полі «прогноз *n*».

Висновки. У роботі обґрунтовано теоретично та підтверджено практично ефективність застосування методів оптимізації структури розрідженої матриці для задач розв'язування СЛАР. Усі підходи базувалися на ідеї зменшення кількості арифметичних операцій шляхом зменшення числа ненульових елементів передобумовлювача. Окрім того, застосування методу паралельних перерізів з наступним відкиданням обрамлення дає змогу побудувати матрицю блочно-діагональної структури, обробку якої надзвичайно легко можна розпаралелити.

В.В. Полянко

ОПТИМИЗАЦИЯ СТРУКТУРЫ РАЗРЕЖЕННОЙ МАТРИЦЫ
ДЛЯ ПОСТРОЕНИЯ ЭФФЕКТИВНОГО ВЫЧИСЛИТЕЛЬНОГО АЛГОРИТМА

Рассматриваются проблемы оптимизации структуры разреженной матрицы для построения эффективного вычислительного алгоритма нахождение решения системы линейных алгебраических уравнений. Предложено несколько подходов для модификации матрицы и их использование в итерационном методе решения системы.

V.V. Polyanko

OPTIMIZATION OF STRUCTURE OF SPARSE MATRIX FOR CONSTRUCTION OF EFFECTIVE COMPUTATION ALGORITM

The problems of optimization of sparse matrix structure for construction of effective computation algorithm for solving system of linear algebraic equations are considered. Some approaches for matrix modifications and its application for iterative method of system solving were proposed.

1. *Обзор* математических пакетов, "позволяющих" решать СЛАУ // Научный форум dxdy. Режим доступу: http://dxdy.ru/topic9158.html.
2. *Хіміч О.М., Полянко В.В., Чистякова Т.В.* Реалізація паралельних алгоритмів методу Гаусса для розріджених матриць // Праці Міжнар. симпозіуму «Питання оптимізації обчислень (ПОО-XXXV)». – К.: Інститут кібернетики ім. В.М. Глушкова НАН України. – 2009. – **2**. – С. 388–393.
3. *Самарский А.А., Николаев Е.С.* Методы решения сеточных уравнений. – М.: Наука, 1978. – 592 с.
4. *Calgaro C., Chehab J.P., Saad Y.* Incremental incomplete LU factorizations with applications to time-dependent PDEs // Report umsi-2008-276, Minnesota Supercomputer Institute, University of Minnesota, Minneapolis, MN, 2008.

5. *Saad Y.* Iterative Methods for Sparse Linear Systems. – PWS Publishing Company, 2000. – 448 p.
6. *Джордж А., Лю Дж.* Численное решение больших разреженных систем уравнений. – М.: Мир, 1984. – 333 с.

Получено 29.03.2010