Obliczenia naukowe

Jakub Kołakowski
nr albumu 221457

Lista 5

Zadanie 1.

Napisać funkcję rozwiązującą układ $Ax = b$ metodą eliminacji Gaussa uwzględniającą specyficzną postać rzadkiej macierzy A dla dwóch wariantów:

a) bez wyboru elementu głównego,
b) z częściowym wyborem elementu głównego

Rozwiązanie:

a) bez wyboru elementu głównego

Parametry wejściowe:
A– kwadratowa, rzadkamacierz A
b–wektor prawych stron b
n – rozmiar kwadratowej macierzy A, oraz wektora prawych stron b
l – rozmiar wszystkich kwadratowych macierzy wewnętrznych (bloków): $A_k, B_k, C_k,$

Parametry wyjściowe:
x – wektor rozmiaru n, rozwiązanie układu $Ax = b$

Algorytm:

Etap 1. Sprowadzenie macierzy A do postaci macierzy trójkątnej górnej. Należy wyzerować niezerowe elementy leżące pod przekątną macierzy.

1. $for\ k = 1\ to\ n - 1$
2. $\quad for\ i = k + 1\ to\ k + l\ -\ k\%l$
3. $\quad\quad wspol\ =\ A[i,k]\ /\ A[k,k]$
4. $\quad\quad \#A[i,k]\ =\ 0.0;$
5. $\quad\quad for\ j = k + 1\ to\ min(n, k + l)$
6. $\quad\quad\quad A[i,j]\ =\ A[i,j]\ -\ wspol\ *\ A[k,j]$
7. $\quad\quad endfor$
8. $\quad\quad b[i]\ =\ b[i]\ -\ wspol * b[k]$
9. $\quad endfor$
10. $endfor$

Opis algorytmu (numer linii + wytłumaczenie):

1. zerowanie elementów będzie się odbywać w n-1 wierszach, oprócz pierwszego wiersza, gdyż nie ma tam elementów pod przekątną macierzy
2. zerujemy elementy w danej kolumnie począwszy od wiersza o 1 niżej od przekątnej, maksymalnie do l wierszy niżej, gdyż niezerowych elementów w bloku jest w danej kolumnie najwyżej l, kolejne ilości elementów do wyzerowania powtarzają się następująco: $l,\ l-1,\ l-2, \dots, 1$
3. zerowanie elementów odbywa się za pomocą odejmowania odpowiedniej wielokrotności wiersza na przekątnej od tych poniżej przekątnej.
4. w tym miejscu moglibyśmy wyzerować dany element poniżej przekątnej, jednak w tym wypadku nie trzeba tego robić, po prostu przyjmujemy, że od tej chwili elementy w danej kolumnie poniżej przekątnej są zerami.

5. musimy zaktualizować wartości komórek w danym wierszu począwszy od komórki o 1 w prawo od tej wyzerowanej aż do mniejszej z wartości: końca tablicy n albo wartości oddalonej o l, gdyż dalej są już zera, aktualizujemy, ponieważ odejmujemy pewną wielokrotność wiersza na przekątnej od tych poniżej.
6. aktualizacja poszczególnych komórek w wierszu, opisana powyżej
8. należy zaktualizować także odpowiednią wartość w wektorze prawych stron b

Analiza złożoności algorytmu z etapu 1:
Pętla zewnętrzna z 1. linijki wykonuje się n-1 razy, więc mamy ograniczenie O(n), pętla wewnętrzna z linijki 2 maksymalnie l razy, więc ograniczenie O(l), pętla najbardziej zagnieżdżona z linijki 5 też maksymalnie l razy, więc O(l). Ostatecznie mamy złożoność O(n * l * l). Jeżeli l zależne jest od n, mamy wtedy złożoność rzędu O(n^3), jeśli jednak jest to jakaś stała wartość to cały etap algorytmu ma złożoność O(n).

Etap 2. Rozwiązanie układu $Ax = b$ gdzie A jest rzadką, trójkątną macierzą górną.

1. $for\ i = n : -1 : 1$
2. $\quad suma = 0;$
3. $\quad for\ j = i + 1 : min(n, i + l)$
4. $\quad\quad suma += A[i,j] * x[j]$
5. $\quad endfor$
6. $\quad x[i] = (b[i] - suma)\ /\ A[i,i]$
7. $endfor$

Opis algorytmu (numer linii + wytłumaczenie):
1. Obliczanie zaczynamy od ostatniego wiersza do pierwszego
3. Na poszczególny wynik x[i] mają wpływ jedynie wartości z macierzy A oddalone o maksymalnie l w prawą stronę macierzy A od przekątnej, gdyż dalej mogą być jedynie zera.
4. Obliczamy sumę $A[i,j] * x[j] + A[i,j+1] * x[j+1] + ... + A[i,j+l] * x[j+l]$
6. Jest to przekształcone równanie $b[i] = suma + x[i] * A[i,i]$

Analiza złożoności algorytmu z etapu 2:
Pętla zewnętrzna z 1. Linijki wykonuje się n razy, więc złożoność O(n), a pętla wewnętrzna z linijki 3. maksymalnie l razy, więc mam O(l). Ostatecznie złożoność to O(n * l). Jeżeli l jest stałą to złożoność jest rzędu O(n), w przeciwnym przypadku O(n^2).

Rozwiązanie:

b) z częściowym wyborem elementu głównego

Parametry wejściowe i wyjściowe takie same jak w rozwiązaniu a.

Algorytm:

Etap 1. Sprowadzenie macierzy A do postaci macierzy trójkątnej górnej. Należy wyzerować niezerowe elementy leżące pod przekątną macierzy. Dla elementów na przekątnej szukamy w tej samej kolumnie największego co do wartości bezwzględnej elementu poniżej przekątnej. Jeżeli takowy znajdziemy to zamieniamy niezerową część wiersza zaczynającą się elementem z przekątnej, z niezerową częścią wiersza, gdzie znaleźliśmy wyżej wymieniony największy element.

1. $for\ k = 1 : n - 1$
2. $\quad max = abs(A[k,k]);$
3. $\quad maxrow = k;$
4. $\quad for\ i = k + 1 : k + l - k\%l$
5. $\quad\quad if(abs(A[i,k]) > max)$
6. $\quad\quad\quad max = abs(A[i,k]);$
7. $\quad\quad\quad maxrow = i;$
8. $\quad\quad end$
9. $\quad end$

10. $if(maxrow > k)$ $\#trzeba\ wykonac\ zamiane\ rzedow: maxrow\ oraz\ k$
11. $it = 1;$
12. $for\ t = k : min(n, k + 2 * l)$ $\#kopiowanie\ 2l + 1\ elementow$
13. $temp[it] = A[k, t];$
14. $it += 1;$
15. $endfor$

$\#zamiana\ elementow\ w\ wektorze\ prawych\ stron$
16. $tempb = b[maxrow];$
17. $b[maxrow] = b[k];$
18. $b[k] = tempb;$

19. $for\ t = k : min(n, k + 2 * l)$
20. $A[k, t] = A[maxrow, t];$
21. $endfor$

22. $it = 1;$
23. $for\ t = k : min(n, k + 2 * l)$
24. $A[maxrow, t] = temp[it];$
25. $it += 1;$
26. $endfor$
27. $endif$
28. $for\ i = k + 1 : k + l - k\%l$
29. $wspol = A[i, k] / A[k, k];$
30. $\#A[i, k] = 0.0;$
31. $for\ j = k + 1 : min(n, k + l + l)$ $\#musimy\ sprawdzac\ dalej\ przez\ przestawienia, o\ l\ dalej$
32. $A[i, j] = A[i, j] - wspol * A[k, j];$
33. $endfor$
34. $b[i] = b[i] - wspol * b[k];$
35. $endfor$
36. $endfor$

Opis algorytmu (numer linii + wytłumaczenie):

4-9. Szukamy największego elementu co do wartości bezwzględnej, który leży w tej samej kolumnie, poniżej danego elementu z przekątnej, w pętli sprawdzamy maksymalnie l wartości, gdyż poniżej w danej kolumnie znajdują się już jedynie zera.

10. jeżeli istnieje rząd z większym co do wartości bezwzględnej elementem to zamieniamy dane wiersze

12-26. zamiana odpowiednich wierszy w macierzy A oraz zamiana odpowiednich elementów z wektorów prawych stron.

12,19,23. w pętli kopiujemy maksymalnie $2l + 1$ elementów

31. Przez to, że wykonywaliśmy przestawienia wierszy, musimy teraz odejmowanie danych wielokrotności wierszy od siebie rozszerzyć o l elementów

Analiza złożoności algorytmu z etapu 1:

Zewnętrzna pętla wykonuje się O(n) razy. Wyszukiwanie największego elementu O(l) razy. Zamiana wierszy O(l), pętla wewnętrzna z linijki 28. O(l) razy oraz najbardziej zagnieżdżona pętla O(l). Ostatecznie cały etap ma złożoność $O(n * l + 3 * n * 2l + n * l * 2l)$ czyli $O(n * l^2)$. Jeżeli l jest stałą to ostatecznie złożoność jest rzędu O(n), a jeśli l jest zależne od n, to O(n^3).

Etap 2. Rozwiązanie układu $Ax = b$ gdzie A jest rzadką, trójkątną macierzą górną.

1. $for\ i = n : -1 : 1$
2. $\quad suma\ =\ 0;$
3. $\quad for\ j = i + 1 : min(n, i + l + l)$
4. $\quad\quad suma\ +=\ A[i,j] * x[j]$
5. $\quad endfor$
6. $\quad x[i]\ =\ (b[i]\ -\ suma)\ /\ A[i,i]$
7. $endfor$

Jedyną zmianą w stosunku do Etapu 2 z punktu a, jest to, że przez wykonywanie zamian wierszy, musimy teraz w pętli, w 3. linijce wykonywać obliczenia o maksymalnie l więcej razy. Złożoność pozostaje więc identyczna jak w etapie 2 z punktu a.

Porównanie zaimplementowanych algorytmów

Dane testowe:

1) Macierz A wygenerowana za pomocą funkcji blockmat z parametrami:
   n = 1000
   l = 4
   ck = 100.0

Algorytm	Gauss	Gauss z częściowym wyborem elementu głównego	x = A\b
Błąd względny	4.72711132180477e-12	2.936977450712567e-15	2.5870912327440445e-15
Czas wykonywania(sekundy)	0.003827	0.005640259	0.002167183

2) Macierz A wygenerowana za pomocą funkcji blockmat z parametrami:
   n = 10000
   l = 4
   ck = 100.0

Algorytm	Gauss	Gauss z częściowym wyborem elementu głównego	x = A\b
Błąd względny	5.200139752213986e-12	3.0307085631215322e-15	5.830480252457435e-14
Czas wykonywania(sekundy)	0.273796	0.365480498	0.026886371

3) Macierz A wygenerowana za pomocą funkcji blockmat z parametrami:
   n = 100 000
   l = 4
   ck = 100.0

Algorytm	Gauss	Gauss z częściowym wyborem elementu głównego	x = A\b
Błąd względny	2.2653386569612032e-12	3.051546815061286e-15	2.6564203092255e-15
Czas wykonywania(sekundy)	46.113316	62.402085323	0.257875487

4) Macierz A wygenerowana za pomocą funkcji blockmat z parametrami:
   n = 10 000
   l = 10
   ck = 100.0

Algorytm	Gauss	Gauss z częściowym wyborem elementu głównego	x = A\b
Błąd względny	1.0084224812678654e-12	3.1514520983759326e-15	2.1130796400872623e-15
Czas wykonywania(sekundy)	1.622544	1.964260159	0.086508742

5) Macierz A wygenerowana za pomocą funkcji blockmat z parametrami:
   n = 10 000
   l = 40
   ck = 100.0

Algorytm	Gauss	Gauss z częściowym wyborem elementu głównego	x = A\b
Błąd względny	9.006247803800444e-12	3.7971867299761065e-15	1.8193093641239756e-15
Czas wykonywania(sekundy)	35.345689	37.614973392	0.215531465

Wnioski:

Ze względu na dodatkowe operacje wyszukiwania odpowiedniego wiersza do zamienienia w Gaussie z częściowym wyborem, metoda ta wykonuje się dłużej niż jej odpowiednik bez częściowego wyboru.
Można zaobserwować, że najszybsza jest wbudowana już eliminacja Gaussa dla macierzy rzadkich. Błąd względny we wbudowanym Gaussie pozostaje praktycznie nie zmieniony niezależnie od zmian parametru wielkości macierzy n, oraz wielkości bloku l i jest on bliski błędowi motody Gaussa z częściowym wyborem elementu głównego. Widać, że Gauss z częściowym wyborem elementu głównego ma mniejszy błąd względny o kilka rzędów wielkości.