Tomasz Beneś
6 stycznia 2020

# Obliczenia Naukowe – Lista 5

## Zadanie 1 – Eliminacja Gaussa

Opis problemu

Nie można użyć standardowej metody ze względu na zbyt duży rozmiar i rzadkość macierzy. Macierz jest regularna, blokowa, z czego każdy rodzaj z trzech bloków tworzy diagonalę. Taka budowa pozwala zoptymalizować algorytm eliminacji Gaussa, by można go użyć do odpowiednio przechowanej macierzy.

Sposób przechowania macierzy

Pamiętanie wszystkich zer w macierzy byłoby bardzo nieefektywne oraz niepotrzebne, dlatego użyta została odpowiednia struktura języka `Julia` - `SparseMatrix`, która zapamiętuje jedynie wartości niezerowe. Ponieważ `SparseMatrix` przechowuje wartości w porządku kolumnowym, a w metodzie eliminacji Gaussa rozważamy przede wszystkim wiersze, przechowywana jest właściwie macierz transponowana. Jako szczegół implementacyjny, nie będzie to brane pod uwagę w dalszych rozważaniach.

Metoda eliminacji Gaussa

Standardowy algorytm eliminacji Gaussa (bez wyboru elementu głównego) zakłada stopniową eliminację zmiennych w odpowiednich wierszach poprzez odejmowanie innych wierszy, aby układ $Ax=b$ zastąpić równoważnym $A'x=b'$ gdzie $A'$ jest macierzą trójkątną górną.

W tym celu, dla $i=1,2,\ldots,n-1$ (gdzie $n$ to rozmiar kwadratowej macierzy $A$ ) odejmujemy od wierszy $A_j$ , $j=i+1,\ldots,n$ krotność $\frac{A_{i,k}}{A_{k,k}}$ wiersza $A_i$ .

Standardowa metoda eliminacji Gaussa ma złożoność $O(n^3)$ .

Modyfikacje

Ze względu na rzadkość macierzy i jej regularną budowę algorytm można przyspieszyć, znacznie redukując ilość wykonywanych operacji, w szczególności tych z zerem.

Diagonalna budowa macierzy zapewnia, że wiele elementów, które w metodzie Gaussa należy zerować, już zerami będzie. Dzięki regularnej strukturze możemy oszacować, który wiersz będzie ostatnim, dla danej kolumny $\rho$ , w którym wartość w tej kolumnie będzie niezerowa.

Ponieważ macierz $B_k$ ma tylko dwie ostatnie kolumny z $l$ kolumn niezerowe, jej wartości wpływać będą jedynie na co $l-1$ i $l$ -tą wartość $LAST\,ROW(\rho)$ . W pozostałych przypadkach, ostatnimi niezerowymi wartościami w kolumnie będą wartości z macierzy $A_k$ . Można więc wyprowadzić wzór:

$$LAST\,ROW(\rho)=min\{l+l\cdot\left\lfloor\frac{\rho+1}{l}\right\rfloor,n\}$$

Zauważmy, że w przypadku $\rho=l-1$ lub $\rho=l$ wartość $\left\lfloor\frac{\rho+1}{l}\right\rfloor=1$ , więc zostaną wzięte pod uwagę wartości z $B_k$ .

W podobny sposób można znaleźć wyrażenie opisujące indeks ostatniej niezerowej kolumny dla danego wiersza $\eta$ . W rozpatrywanej macierzy wartość ta zawsze znajdzie się w diagonalnej macierzy $C_k$ . Zauważmy, że te są zawsze odległe o $l$ od diagonali macierzy $A$ , stąd mamy prosty wzór:

$$LAST\,COLUMN(\eta)=min\{\eta+l,n\}$$

Warto zauważyć, że w wyniku kolejnych kroków eliminacji Gaussa nie pojawią się żadne niezerowe wartości powyżej diagonali macierzy $C_k$ , więc wartość $LAST\,COLUMN(\eta)$ pozostanie aktualna do końca.

Cały zmodyfikowany algorytm wygląda następująco:

```
for k = 1 to n-1
      lastrow = LAST ROW(k)
      lastcol = LAST COLUMN(k)
      for i = k+1 to lastrow
            z = A[i,k] / A[k,k]
            A[i,k] = 0
            for j = k+1 to lastcol
                  A[i,j] -= z· A[k,j]
            b[i] -= z· b[k]
```

Zakładając, że $l$ jest stałe, złożoność zmodyfikowanego algorytmu wynosi $O(n)$ , ponieważ zewnętrzna pętla wykonuje $n-1$ przebiegów, a wewnętrzne pętle odpowiednio maksymalnie $2l$ i $l$ .

Algorytm podstawiania wstecz

Aby z otrzymanego w wyniku eliminacji Gaussa układu trójkątnego otrzymać wektor $x$ , należy wykonać jeszcze jeden krok, jakim jest algorytm podstawiania wstecz. Algorytm jest dość trywialny, by nie przywoływać go w formie pseudokodu. Zauważmy jednak, że dzięki znajomości $LAST\,COLUMN(\eta)$ możemy ograniczyć liczbę operacji dodawania do $l$ . Jeśli $l$ jest stałe, ze złożoności $O(n^2)$ schodzimy do $O(n)$ .

Metoda eliminacji Gaussa z częściowym wyborem elementu głównego

Standardowa metoda może nie działać (lub działać nieprawidłowo w arytmetyce zmiennoprzecinkowej) w pewnych szczególnych przypadkach:

1. Na diagonali macierzy $A$ pojawia się zero.

2. Wartość na diagonali macierzy $A$ jest bardzo mała.

Aby sprostać takim macierzom, stosuje się częściowy wybór elementu głównego. Zmiana polega na wyborze w $i$ -tym kroku takiego wiersza, dla którego wartość w $i$ -tej kolumnie jest największa.

Ponieważ zamiana wierszy jest czasowo nieefektywna, tworzony jest wektor permutacji wierszy $\pi$ , który przechowuje aktualny indeks $i$ -tego wiersza w macierzy $A$ , tak, że prawdziwy $i$ -ty wiersz znajduje się w pamięci pod $A_{\pi_i}$ .

Częściowy wybór elementu głównego sprawia również, że wiersze mogą zostać odjęte w innej kolejności, przez co tracona jest stała właściwość $LAST\,COLUMN(\eta)$ . Potrzebne jest więc szersze oszacowanie. Ponieważ od $i$ -tego wiersza odejmować będziemy tylko te wiersze, które w mają w choć jedną „wspólną” kolumnę (tj. taką, że w obu wierszach w danej kolumnie jest wartość niezerowa), możemy policzyć, jak najdalej mogą znajdować się dwa wiersze, które mogą być od siebie od siebie odejmowane. Stąd wzór:

$$LAST\,COLUMN'(\eta)=min\{2l+l\cdot\left\lfloor\frac{\eta+1}{l}\right\rfloor,n\}$$

Zmianie musi ulec również algorytm podstawiania wstecz, biorąc pod uwagę nowe $LAST\,COLUMN'(\eta)$ oraz wektor permutacji wierszy $\pi$ . Nie wpływa to jednak na jego złożoność dla stałego $l$ .

Poniższy algorytm przedstawia zmodyfikowany algorytm z częściowym wyborem elementu głównego:

$\pi$ = $\{1,2,\ldots,n\}$
`for k = 1 to n-1`
    `lastrow =` $LAST\,ROW(k)$
    `lastcol =` $LAST\,COLUMN'(k)$
    `for i = k+1 to lastrow`
        `r =` $m$ `such that` $A_{\pi_m,k}=max(|A_{\pi_q,k}|:q\in\{i,\ldots,lastrow\})$
        $\pi_k,\pi_r$ = $\pi_r,\pi_k$
        `z =` $\dfrac{A_{\pi_i,k}}{A_{\pi_k,k}}$
        $A_{\pi_i,k}$ `= 0`
        `for j = k+1 to lastcol`
            $A_{\pi_i,j}$ `-= z·` $A_{\pi_k,j}$
        $b_{\pi_i}$ `-= z·` $b_{\pi_k}$

<u>Analiza czasu wykonania i zużycia pamięci</u>

Poniższe wykresy przedstawiają potrzebny czas i pamięć dla macierzy o $n=\{1000,2000,\ldots,50000\}$ . Można zauważyć, że dla czasu, wbrew oczekiwaniom, nie wykres nie jest liniowy. Jest to związane z czasem dostępu do elementów `SparseMatrix`. Zamieszczone są również wykresy zawierające błędy względne każdego z eksperymentów. Można zauważyć, że w przypadku wyboru elementu głównego precyzja znacząco wzrasta, co oznacza, że algorytm faktycznie rozwiązuje wspomniany problem (2), oraz że wielkość $n$ nie wpływa na wielkość błędu.

We wszystkich eksperymentach przyjęte jest $l=4$ i wskaźnik uwarunkowania macierzy $A_k=10$. Macierze generowane były funkcją `blockmat` autorstwa prof. Pawła Zielińskiego.