Obliczenia naukowe

Sprawozdanie

Lista 4

Mateusz Laskowski

06.01.2019

1. **Zadanie 1**
   1. **Opis problemu**

Zaimplementuj metodę rozwiązującą układ metodą eliminacji Gaussa uwzględniając specyficzną postać macierzy . Macierz mamy obliczyć dla dwóch wariantów:

* bez wyboru elementu głównego
* z częściowym wyborem elementu głównego

gdzie wektor prawych stron to

Macierz ma specyficzną postać, która jest przedstawiona poniżej:

Gdzie przy założeniu, że jest podzielne przez , gdzie jest rozmiarem wszystkich kwadratowych macierzy wewnętrznych (bloków): . Bloki

, jest macierzą gęstą, 0 jest kwadratową macierzą zerową stopnia . Macierz , jest następującej postaci:

ma tylko jedną, ostatnią kolumnę niezerową. Natomiast blok ,

jest macierzą diagonalną:

Podczas implementacji trzeba wziąć pod uwagę, że złożoność algorytmu nie może wynosić jak w standardowym algorytmie eliminacji Gaussa, czyli *,* lecz *.*

* 1. **Analiza eliminacji Gaussa**

Metodę eliminacji Gaussa można podzielić na dwa oddzielne etapy. Pierwszy z nich to doprowadzenie macierzy do postaci schodkowej, czyli taka postać, gdzie niezerowe komórki znajdują się jedynie powyżej komórek o indeksie wiersza i kolumny równym sobie. Drugim etapem w metodzie to rozwiązanie odpowiadającej tej macierzy układu równań. Poniżej przykład macierzy, którą chcemy uzyskać, czyli macierz schodkowa:

* 1. **Opis rozwiązania**
     1. **Eliminacja Gaussa bez wyboru elementu głównego**

Pierwszy etap algorytmu to manipulacja macierzą – za pomocą operacji elementarnych – w celu uzyskania macierzy schodkowej. Operacja tego działania opiera się na wyborze wiersza głównego, następnie mnożeniu go przez odpowiedni czynnik – oznaczony – i dodawaniu do każdych kolejnych wierszy tak, aby wiersze te w danej kolumnie zostały wyzerowane.

W tym wypadku za wiersz główny został wybrany wiersz , a przy każdym innym wierszu współczynnik jest równy . Po wykonaniu wszystkich działań na kolejnych postaciach macierzy , otrzymując macierz trójkątną górną, wcześniej opisaną jako macierzą schodkową.

Mając już taką macierz trójkątną można przejść do etapu drugiego, czyli obliczania szukanego wektora. Zaczynając od ostatniego wiersza ostatniej kolumny można wyliczyć szukany wektor *x* rozwiązań równania.

* + 1. **Eliminacja Gaussa z wyborem elementu głównego**

Algorytm wykonuje wszystkie operacje jak w powyższym opisanym schemacie działania, lecz trzeba dodać dodatkowe wstępne założenie, gdzie komórki *,* które są używane w metodzie eliminacji Gaussa jako dzielniki nie mogą być równe zero. W przeciwnym wypadku dochodziłoby do dzielenia przez zero. Na dodatek nasz trzeba pamiętać, że wartości nie mogą być bliskie zera, ponieważ zostaną zaokrąglone do zera. Na straży oscylowaniu przy zerze będzie stał macheps, gdzie przy przyrównaniu z nim zwróci błąd i zakończy dalszemu wyliczaniu. Doprowadzenie macierzy do postaci, w której komórki mają jak największą wartość, co pozwala poprawić numeryczną dokładność późniejszych wyliczeń. Więc szukam wiersza, spośród wierszy takich, że *.* Wybór spośród wierszy zaburzyłby strukturę macierzy, do której dążymy, czyli macierzy schodkowej, a dokładniej zerową dolną lewą macierz trójkątną. Przykładowe działanie powyżej opisanego schematu:

Wybierana jest komórka o największej wartości spośród , gdzie , a następnie zamienione zostają miejscami wiersze z wierszem .

* 1. **Złożoność algorytmu**

Przy podanej w zadaniu macierzy, w przypadku normalnego działania eliminacji Gaussa, osiągnęlibyśmy złożoność *.* Zadanie polegało na tym, aby zminimalizować złożoność algorytmu do złożoności liniowej oraz osiągnąć jak najlepszą złożoność pamięciową. W celu osiągnięcia jak najmniejszej złożoności pamięciowej, do przechowywania wszelkich macierzy zostały użyte macierze typu SparseMatrixCSC w bibliotece języka Julia. Tego typu macierze są wykorzystywane w macierzach rzadkich, ponieważ zapamiętuje jedynie elementy niezerowe podawane do komórek. Aby móc osiągnąć złożoność liniową w eliminacji Gaussa na podanym schemacie macierzy należy wziąć jedynie komórki z wartościami niezerowymi, które znajdują się w blokach . Aby tak uczynić, algorytm zawęża w każdej iteracji zakres wierszy do wyzerowania kolumny do *,*ponieważ taka jest maksymalna rozpiętość niezerowych wartości. Natomiast maksymalny zakres kolumn w danym wierszu może być do . Takimi oto sprytnymi sposobami otrzymujemy złożoność , pamiętając, że jest stałą, czyli niezależną od , co w notacji duże *O* daje nam to liniową złożoność.

* 1. **A**