

Sprawozdanie z listy 5

Jarosław Socha

268463

3 stycznia 2023

1 Eliminacja Gaussa

Eliminacja Gaussa jest metodą rozwiązywania układów równań. Jeśli zastosujemy formę macierzową do reprezentacji układu równań, gdzie A to macierz ograniczeń, $x = [x_1, \dots, x_n]$ to wektor niewiadomych, a b to wektor prawych stron, to metoda pozwala nam rozwiązać równanie

$$Ax = b$$

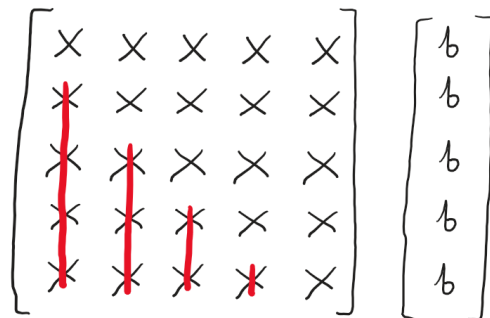
Będziemy to robić częściowo zerując kolumny macierzy, tak aby najpierw otrzymać macierz trójkątną górną, a następnie diagonalną.

Algorytm:

1. Macierz trójkątna górna

Weźmy pierwszy wiersz macierzy. Wyznamy dla każdego wiersza i pod tym wierszem jego mnożnik $I_{i,1}$, po czym od każdego z tych wierszy odejmujemy wiersz pierwszy pomnożony przez $I_{i,1}$ (oznacza to odjęcie od każdego elementu $A[i, k]$ iloczynu $A[1, k] \cdot I_{i,1}$). W ten sposób wyzerowaliśmy wszystkie elementy pierwszej kolumny leżące poniżej przekątnej (czyli elementu $A[1, 1]$). Zauważmy, że jeśli nie będziemy już odejmować od niczego elementów wiersza pierwszego, to otrzymane zera pozostaną zerami. Powtarzając ten proces dla kolejnych wierszy (zerując zawsze wiersze poniżej), otrzymamy macierz trójkątną górną.

W momencie gdy odejmujemy wiersz od wiersza, odejmujemy też odpowiadające tym wierszom prawe strony równania, czyli elementy wektora b , pomnożone przez odpowiadający mnożnik.


$$\begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{bmatrix} \begin{bmatrix} b \\ b \\ b \\ b \\ b \end{bmatrix}$$

Rysunek 1: \times to wartości. Czerwone kreski symbolizują zerowanie części kolumn w kolejnych krokach

2. Macierz diagonalna

Teraz wykonamy tę samą procedurę, ale od dołu, zaczynając od prawej strony macierzy. Warto zauważyć, że nie trzeba tu odejmować całego wiersza, wystarczy jeden jego element, gdyż w każdym kroku wiersz główny, który rozpatrujemy, będzie miał tylko jeden element niezerowy (jest to element leżący na przekątnej macierzy). Tak jak poprzednio, jeśli odejmujemy wiersze (bądź tak jak tutaj tylko elementy) od siebie, to robimy tak samo z elementami wektora b , bo odpowiada to równaniom, które odejmujemy stronami. Po przejściu w ten sposób po wszystkich elementach przekątnej, niewyzerowane zostaną jedynie elementy na przekątnej macierzy

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & \times \end{bmatrix} \begin{bmatrix} b \\ b \\ b \\ b \\ b \end{bmatrix}$$

Rysunek 2: 0 to zera. Czerwone kreski symbolizują zerowanie części kolumn w kolejnych krokach

3. Wynik

Teraz wystarczy każdy wiersz (czyli w tym momencie element na przekątnej i element macierzy b) podzielić przez element na przekątnej. Otrzymamy w ten sposób macierz jednostkową i macierz wartości b , co odpowiada wynikowi

$$x_1 = b[1] \quad x_2 = b[2] \quad \dots \quad x_{n-1} = b[n-1] \quad x_n = b[n]$$

Czyli w macierzy b leżą wartości zmiennych spełniające układ równań $Ax = b$.

Ten krok można zawrzeć w kroku drugim, wykonując to dzielenie zanim zrobimy nowy mnożnik.

$$\begin{bmatrix} \times & 0 & 0 & 0 & 0 \\ 0 & \times & 0 & 0 & 0 \\ 0 & 0 & \times & 0 & 0 \\ 0 & 0 & 0 & \times & 0 \\ 0 & 0 & 0 & 0 & \times \end{bmatrix} \begin{bmatrix} b \\ b \\ b \\ b \\ b \end{bmatrix}$$

Rysunek 3: Wartości X po tym kroku zmieniają się w jedynki, dając nam rozwiązanie

Złożoność obliczeniowa metody eliminacji Gaussa to $O(n^3)$.

2 Rozkład LU

Rozkład LU to reprezentacja macierzy A za pomocą dwóch macierzy L i U (odpowiednio trójkątnej dolnej i trójkątnej górnej) taka, że $A = LU$. Taki rozkład w kontekście rozwiązywania układów równań pozwala na wykonanie części obliczeń przed poznaniem prawych stron, co przydaje się w niektórych zastosowaniach. Dodatkowo, jako że do wykonania części obliczeń nie jest potrzebny wektor prawych stron, to możemy raz policzony rozkład wykorzystać wiele razy przy różnych prawych stronach.

Aby efektywnie pamiętać rozkład, wykorzystujemy macierz tych samych wymiarów co macierz A . W jej lewym dolnym trójkącie pamiętamy elementy macierzy L , a w prawym górnym elementy macierzy U . Jedyny konflikt, gdzie obie wartości są niezerowe, mógłby wystąpić na przekątnej. Wiemy jednak, że przekątna macierzy L składa się z samych jedynek, więc nie musimy zapamiętywać tych wartości i przekątna jest częścią macierzy U .

$$\left| \begin{array}{cccc} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{array} \right| \left| \begin{array}{cccc} u & u & u & u \\ 0 & u & u & u \\ 0 & 0 & u & u \\ 0 & 0 & 0 & u \end{array} \right| \rightarrow \left| \begin{array}{cccc} u(1) & u & u & u \\ 1 & u(1) & u & u \\ 1 & 1 & u(1) & u \\ 1 & 1 & 1 & u(1) \end{array} \right|$$

Efektywne pamiętanie rozkładu LU

Algorytm:

1. Rozkład LU

Postępujemy tutaj tak jak w pierwszej części eliminacji Gaussa (sekcja 1 w poprzednim algorytmie), przy czym nie modyfikujemy wektora prawych stron b , ponieważ ta część obliczeń ma być od niego niezależna. Dodatkową modyfikacją jest to, że gdy obliczymy mnożnik $I_{i,k}$ to wpisujemy go w macierz na pozycji $[i,k]$ - będzie on częścią macierzy L . Po wykonaniu operacji dla wszystkich wierszy otrzymamy optymalnie zapamiętany rozkład LU .

2. Wynik

Gdy dostaniemy wektor prawych stron i mamy rozkład LU , to aby otrzymać wynik, musimy rozwiązać dwa podproblemy: $Ly = b$, a następnie $Ux = y$, co sumarycznie rozwiązuje $Ax = b$.

(a) $Ly = b$

Bierzemy pierwszy element wektora b i pierwszą kolumnę macierzy L . Od każdego elementu $b[i]$ będziemy odejmować iloczyn $L[i,1] \cdot b[1]$. Następnie powtarzamy proces dla następnych elementów, zawsze iterując przez elementy niżej od głównego. Pod koniec sytuacja będzie równoważna takiej, gdzie po lewej jest macierz jednostkowa (nie zerujemy elementów, aby móc wielokrotnie używać macierzy LU), więc wektor b jest rozwiązaniem.

Można zauważyć, że te operacje odpowiadają pierwszej części eliminacji Gaussa dla wektora prawych stron, ale są szybsze ze względu na policzone wcześniej mnożniki.

(b) $Ux = y$

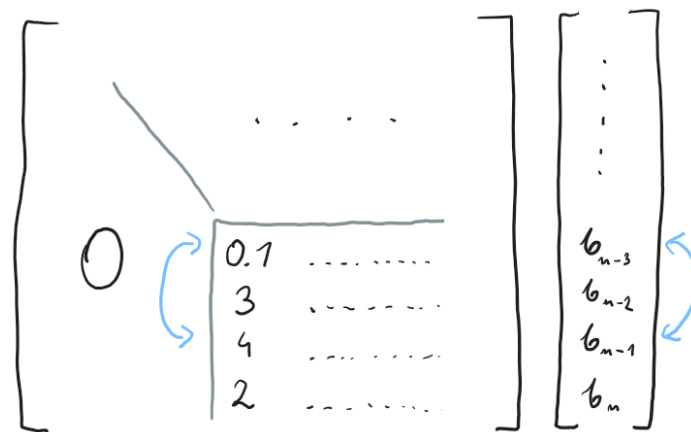
Ten krok wygląda dokładnie tak samo jak drugi i trzeci krok eliminacji Gaussa. Tak jak poprzednio, nie odejmujemy elementów macierzy LU bo nie jest to potrzebne, a dodatkowo pozwala na zachowanie struktury macierzy w celu obliczania wyniku dla kolejnych prawych stron.

Po wykonaniu tej części w wektorze b są wyniki równania $Ax = b$.

Choć złożoność tej metody to także $O(n^3)$, to gdy mamy już rozkład LU i dostaniemy prawą stronę, to koszt rozwiązania to tylko $O(n^2)$, więc metoda świetnie się sprawdzi przy wielu prawych stronach do rozwiązania.

3 Częściowy wybór

Podczas któregoś z algorytmów może zdarzyć się, że element na przekątnej jest bliski zeru. Przy obliczeniach dzielimy wtedy przez element bliski zeru i wynik może być silnie zaburzony. Jedną z metod pomagającą temu zapobiec jest częściowy wybór elementu głównego. W każdym kroku pierwszej części metody eliminacji Gaussa lub podziału LU , zanim wybierzemy element główny, który będzie mianownikiem mnożnika, sprawdzamy, czy poniżej w jego kolumnie nie ma elementu większego co do wartości bezwzględnej. Jeśli jest, to zamieniamy ze sobą wiersz główny ze znalezionym (a także odpowiadające im wartości w wektorze b), co sprawi, że mamy duże szanse ominąć niebezpieczne wartości bliskie zeru. Wynik po takiej zamianie wciąż będzie prawidłowy, ponieważ taka operacja odpowiada zamianie kolejności równań w układzie równań, co nie zaburza wyniku.



Operacja zamiany wierszy ze sobą (niebieskie strzałki) na podstawie wartości elementów głównych

Choć taka operacja wymaga dodatkowych obliczeń, to otrzymany wynik nie będzie gorszy od tego, który uzyskalibyśmy nie używając częściowego wyboru. Nie tylko zabezpieczamy się przed dużym błędem w przypadku patologicznym, ale także zyskujemy mniejszy błąd obliczeń w innych przypadkach.

4 Implementacja biblioteki

Celem poniższych sekcji jest opisanie optymalnej implementacji biblioteki z powyższymi algorytmami dla specyficznej macierzy do obliczeń chemicznych.

4.1 Postać macierzy

$$\begin{vmatrix} A & C & 0 & 0 & 0 & \dots & 0 \\ B & A & C & 0 & 0 & \dots & 0 \\ 0 & B & A & C & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & \dots & 0 & B & A & C & 0 \\ 0 & \dots & 0 & 0 & B & A & C \\ 0 & \dots & 0 & 0 & 0 & B & A \end{vmatrix}$$

Powyżej rozpisana jest struktura macierzy, w której wszystkie widoczne elementy to specyficzne macierze kwadratowe wymiaru $l \times l$ (l dzieli n). 0 to macierz kwadratowa zer, A to kwadratowa macierz gęsta, B to macierz, która elementy niezerowe posiada tylko w ostatniej kolumnie, a C to macierz przekątniowa.

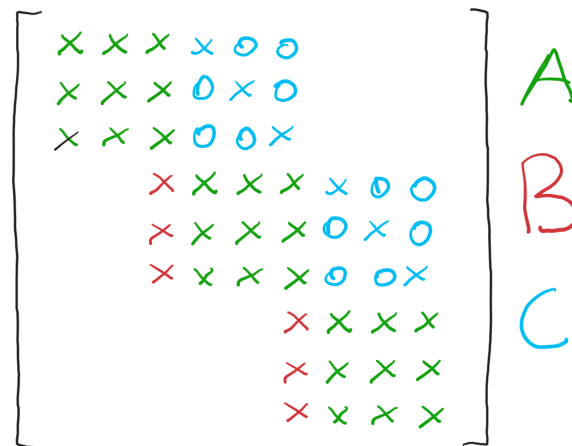
4.2 Pamiętanie macierzy

Jako że podana macierz jest rzadka, nie trzeba przetrzymywać jej zer w pamięci. Konkretniej, jako że chcemy wykonywać na niej obliczenia, możemy nie pamiętać elementów, które na pewno nigdy w wyniku obliczeń nie zostaną zmodyfikowane, czyli zawsze będą zerami.

Zauważmy, że w pierwszej części metody eliminacji Gaussa, jeśli pierwszy element w wierszu pod elementem głównym jest równy zero, to mnożnik I jest równy zero, więc niczego nie odejmujemy od wiersza, zatem zera zostają zerami. Za to w podziale LU, jeśli mnożniki są zerami, to elementy macierzy też. Oznacza to, że dla każdego wiersza nie musimy pamiętać wszystkich jego zer, które poprzedzają pierwszy niezerowy element. Dla elementów po prawej stronie od blokowej przekątnej, jeśli elementy nad danym elementem są zerami, to w wyniku operacji algorytmu ten element nie zmienia wartości, czyli elementy będące zerami i mające nad sobą same zera się nie zmieniają. Sytuacja wygląda analogicznie w drugiej części algorytmu, tutaj zera mające na prawo od siebie same zera, jak i takie, które mają pod sobą same zera, nie zmieniają wartości. Zbiory niezmiennych zer w obydwu krokach się pokrywają, a więc nie musimy ich pamiętać. Zostajemy z trzema rodzajami struktur do zapamiętania:

- Ostatni wiersz macierzy B - zapamiętujemy w wektorze liczb
- Macierz A - zapamiętujemy w macierzy liczb
- Dolny trójkąt macierzy C - zapamiętujemy jako macierz liczb, ze względu na częściowy wybór, o czym w późniejszym rozdziale.

Każda z tych struktur wystąpi odpowiednio $v - 1$, v i $v - 1$ razy ($v = \frac{n}{l}$), a więc zapamiętujemy je w wektorach takich rozmiarów.



Rysunek 4: Struktura macierzy zapamiętana w podstrukturach według kolorów. **X** to komórki z wartościami niezerowymi, a **0** to komórki, które mogą nie być zerami wraz z działaniem algorytmów

Do struktury dodałem funkcję mapującą, która sprawia, że możemy dostać się do macierzy za pomocą rzeczywistych wartości wiersza i kolumny, a funkcja dostanie się do odpowiednich elementów struktury. W funkcji sprawdzane są maksymalnie trzy warunki i wykonywanych jest kilka obliczeń, więc dostęp jest w czasie stałym.

Rozkład LU jest efektywnie pamiętany w tej samej strukturze (lewa część to elementy L , a prawa to elementy U).

4.3 Algorytmy

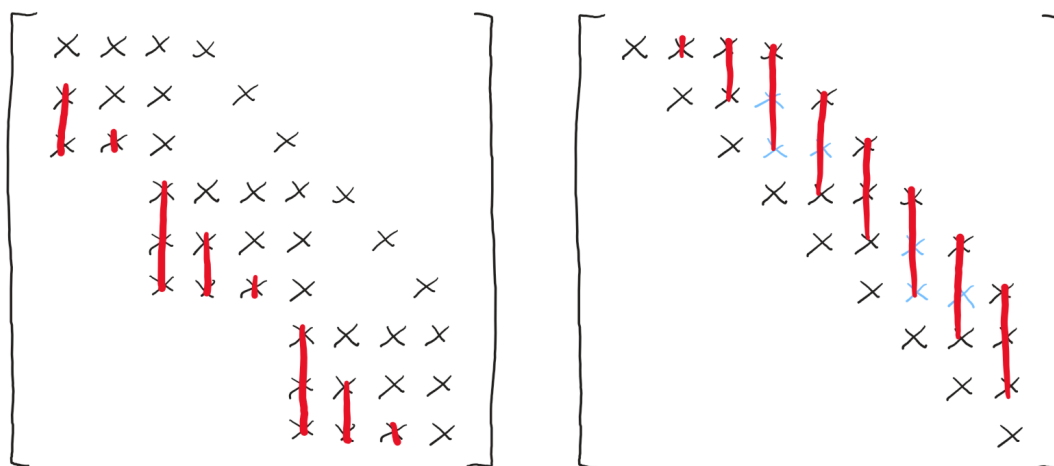
W eliminacji Gaussa dla tej macierzy M jesteśmy w stanie ograniczyć pętle do bloków, na których działają. Pętle pierwszego kroku:

1. Dla każdego elementu przekątnej $M[m, m]$
2. Dla każdego wiersza $i < m$
3. Dla każdego elementu wiersza i o drugim indeksie większym niż m

Ze względu na rzadkość macierzy możemy ograniczyć w następujący sposób:

1. Dla każdego elementu przekątnej $M[m, m]$
2. Dla każdego wiersza $i < m$ ale w tym samym bloku co m ¹
3. Dla każdego elementu wiersza i o drugim indeksie większym niż element m aż do ostatniego niezerowego elementu wiersza m

¹Wyjątek od tej reguły jest gdy m jest końcem sekcji, wtedy do pętli bierzemy także następną sekcję ze względu na strukturę B



Rysunek 5: Kroki eliminacji Gaussa dla przykładowej macierzy $n = 9$, $l = 3$

Choć pierwsza pętla wykona się n razy w obu przypadkach, to druga zamiast maksymalnie n razy wykona się już tylko maksymalnie l razy, a trzecia zamiast n razy wykona się maksymalnie $2l$ razy, co daje nam spadek złożoności obliczeniowej z $O(n^3)$ do $O(n \cdot l^2)$, a jeśli przyjmiemy, że l jest stałą, to złożoność spada do liniowej $O(n)$.

Drugi krok staje się jeszcze prostszy, ponieważ za każdym razem odejmujemy tylko jeden element, czyli działamy na jednej kolumnie. Dzięki przekątniowości macierzy C , oznacza to l operacji, czyli ta część też jest liniowa.

Dla rozkładu LU pierwsza część wygląda analogicznie do metody eliminacji Gaussa. W drugiej części, zanim zrobimy to co robi druga część eliminacji Gaussa, musimy dostosować wektor prawych stron b . Dla każdego elementu przekątnej wykonamy maksymalnie l operacji (każdy taki element ma maksymalnie l elementów niezerowych pod sobą), co daje nam złożoność liniową, czyli cały rozkład ma złożoność $O(n)$.

4.4 Częściowy wybór

Częściowy wybór to modyfikacja algorytmów, która w każdym kroku będzie mogła zamieniać dwa wiersze ze sobą. Aby nie musieć zamieniać wartości ze sobą (co zabiera czas i moc obliczeniową), dodajemy do struktury macierz permutacji w postaci wektora wierszy P , gdzie $P[i]$ przetrzymuje wartość indeksu wiersza, który w tym momencie znajduje się na i -tym miejscu. Gdy częściowy wybór zamieni ze sobą dwa wiersze, to zamieniamy miejscami tylko dwie wartości w macierzy permutacji, a gdy w dalszych obliczeniach chcemy dostać się do jakiegoś wiersza to przepuszczamy indeks przez wektor P .

Dodatkowo częściowy wybór ograniczamy do sekcji, czyli szukamy maksymalnego elementu, ale tylko w danym bloku. Jeśli nie byłoby tego ograniczenia, to w patologicznym przypadku mogłoby dojść do tego, że ostatni wiersz sekcji k zostałby zamieniony z ostatnim wierszem sekcji $k + 1$, a jeśli zdarzyłoby się to kilka razy, to implementacja używająca wektora permutacji by nie zadziałała, ponieważ jeden wiersz musiałby przetrzymywać wszystkie wartości aż do końca macierzy, na wypadek gdyby został zamieniony wielokrotnie. Oznacza to, że z każdą zamianą musielibyśmy zamieniać wszystkie elementy wierszy miejscami. Korzyści, jakie przynosi włączenie innych sekcji do częściowego wyboru, są mniejsze od tych, które daje nam przyspieszenie obliczeń macierzą permutacji.

4.5 Dodatkowe funkcjonalności biblioteki

Biblioteka umożliwia policzenie dla macierzy wyniku metodą eliminacji Gaussa, obliczenia rozkładu LU oraz obliczenia wyniku używając tego rozkładu. Dodatkowo daje możliwość wczytania macierzy z pliku, wczytania wektora prawych stron z pliku, i wypisania wyniku do pliku. Dodatkową funkcjonalnością jest możliwość sprawdzenia błędu metody dla danej macierzy. Tworzymy na podstawie macierzy A wektor prawych stron b , aby wynikiem poprawnym był wektor jedynek. Możemy to osiągnąć podstawiając za $b[i]$ sumę współczynników w wierszu i macierzy. Następnie procedura liczy wynik oraz błąd względny między poprawnym wynikiem (same jedynki) a otrzymanym, po czym wpisuje ten błąd do pliku razem z wynikiem obliczeń.

4.6 Testy biblioteki na przykładowych macierzach

4.6.1 Błąd metody

n	Bez wyboru	Częściowy wybór
16	$6.905730 \cdot 10^{-15}$	$4.775249 \cdot 10^{-16}$
10000	$5.124567 \cdot 10^{-14}$	$3.559160 \cdot 10^{-16}$
50000	$9.245601 \cdot 10^{-14}$	$4.080953 \cdot 10^{-16}$
100000	$6.031082 \cdot 10^{-14}$	$3.045086 \cdot 10^{-16}$
300000	$5.058324 \cdot 10^{-13}$	$4.026873 \cdot 10^{-16}$
500000	$1.359832 \cdot 10^{-13}$	$4.013795 \cdot 10^{-16}$

Tabela 1: Błędy względne dla testowych macierzy o różnych wielkościach

Jak widać, w każdym przypadku częściowy wybór poprawia wynik, ale i tak wszystkie błędy są na niskim poziomie, co dowodzi poprawności i użyteczności algorytmu.

4.6.2 Złożoność czasowa

Choć rozkład LU , rozwiązanie rozkładu LU i eliminacja Gaussa wszystkie mają liniową złożoność obliczeniową, to w analizie czasowej współczynnik ma znaczenie, co pokazuje poniższa tabela.

n	Gauss	Gauss z wyborem	LU	rozwiązanie LU	LU z wyborem	rozwiązanie LU z wyborem
100000	1.048 s	1.474 s	1.031 s	0.495 s	1.262 s	0.451 s
300000	2.791 s	4.000 s	2.116 s	1.113 s	2.782 s	1.037 s
500000	5.483 s	6.961 s	4.338 s	2.824 s	5.565 s	6.356 s

Tabela 2: Czas wykonywania algorytmów dla macierzy wybranych wielkości

Z eksperymentów wynika, że algorytmy są szybkie ze względu na optymalną implementację algorytmu oraz postaci macierzy. W przypadku jednorazowego obliczania rozwiązania dla macierzy, czas eliminacji Gaussa jest mniejszy od sumarycznego kosztu rozkładu i rozwiązania LU , więc należy wybrać eliminację Gaussa. Za to jeśli mamy kilka prawych stron do obliczenia, to bardziej opłaca się rozłożyć macierz na postać LU . Dodatkowo zastosowanie częściowego wyboru nie zwiększa bardzo znacząco czasu wykonania, a daje lepsze wyniki, więc warto go stosować.