

## **Wykład III**

### **Układy równań liniowych i dekompozycje macierzy**

- Metody eliminacji i podstawienia wstecz
- Metoda dekompozycji LU i jej zastosowania
- Metody dla macierzy specjalnych i rzadkich
- Metody iteracyjne

## Problem do rozwiązania

Dla równania macierzowego (reprezentującego układ równań liniowych)  $\mathbf{Ax} = \mathbf{b}$ ,  $\rightarrow \mathbf{x}=?$  gdzie  $\mathbf{A}$  (macierz kwadratowa  $N \times N$ ) i  $\mathbf{b}$  (wektor  $N \times 1$ ) są zadane, a  $\mathbf{x}$  (wektor  $N \times 1$ ) jest niewiadomą, należy znaleźć zawartość wektora  $\mathbf{x}$  spełniającą ten układ równań. Graficzna interpretacja zadania to problem znalezienia punktu wspólnego  $N$  hiperpłaszczyzn w przestrzeni  $N$ -wymiarowej. Dla  $N=2$  to punkt przecięcia dwóch prostych na płaszczyźnie. Interpretacja wektorowa to problem znalezienia współczynników  $\mathbf{x}$  kombinacji liniowej wektorów bazowych (kolumn macierzy  $\mathbf{A}$ ), dającej w wyniku wektor  $\mathbf{b}$ .

W zależności od docelowej architektury i wymagań czasowych zadanie należy rozwiązać szybko (aplikacje *real-time*) i z małym wykorzystaniem pamięci (procesory sterujące, procesory DSP).

### Przykład: Miksowanie dźwięków z wielu źródeł

Wnętrze samochodu jest hałaśliwym otoczeniem. Odgłosy silnika, szum powietrza za oknem, oraz pasażerowie na tylnych siedzeniach przeszkadzają kierowcy w prowadzeniu rozmowy przez telefon w czasie jazdy. Rozmówca na drugim końcu łącza słyszy dźwięki ze wszystkich źródeł ale jest zainteresowany tylko głosem kierowcy. Rozwiązaniem może być układ wielu mikrofonów, do których poszczególne dźwięki docierają z różnym natężeniem. W takim przypadku  $\mathbf{A}$  to macierz miksowania wektora dźwięków źródłowych  $\mathbf{x}$ , zaś  $\mathbf{b}$  to wektor dźwięków wynikowych na wyjściu mikrofonów. Odwracając proces sumowania dźwięków (miksowania) przez rozwiązanie równania, możemy odzyskać wektor dźwięków źródłowych, w tym głosu kierowcy.

## Rozwiązywanie równania macierzowego metodami eliminacji

Druga ze „szkolnych” metod rozwiązywania układów równań liniowych (obok metody wyznaczników Cramera), wspominanych na poprzednim wykładzie, to metoda eliminacji zmiennych poprzez wydzielenie zmiennej w danym równaniu i podstawienie do innego równania. Taka eliminacja określona w precyzyjny algorytmiczny sposób to **metoda Gaussa-Jordana**, prowadzącą do **diagonalizacji macierzy A** (rozdzielenia poszczególnych zmiennych) i **metoda eliminacji Gaussa**, przekształcająca macierz **A** do postaci trójkątnej górnej i układu równań łatwo rozwiązywanego metodą iteracyjną. Nie należy jednak traktować metod eliminacji jako podstawowych – służą one przede wszystkim jako przykład **dydaktyczny** i wprowadzenie do **metod dekompozycji** macierzy.

## Eliminacja w działaniu

Działanie metod eliminacji opiera się na dwóch podstawowych operacjach, które prowadzą do równoważnego (wg rozwiązania) układu równań, ale w postaci dużo prostszej obliczeniowo:

- zastąpienie jednego z równań kombinacją liniową tego równania z innym z równań,
- mnożenie jednego z równań przez stałą (szczególny przypadek poprzedniej operacji).

W celu uodpornienia metod eliminacji na dzielenie przez zero i na zaburzenie współczynników układu równań stosowane są dodatkowo operacje zamiany wierszy i kolumn układu równań, prowadzące do ustalenia takich wartości elementów diagonalnych macierzy, które są najlepsze dla dokładności obliczeń.

### Eliminacja Gaussa-Jordana

W każdym  $k$ -tym kroku algorytmu eliminacji Gaussa-Jordana wykonujemy dwie operacje:

- 1) ustawienie elementu diagonalnego  $a_{k,k}$  macierzy  $\mathbf{A}$  na wartość 1 przez dzielenie współczynników  $k$ -tego równania (wiersza)  $\{a_{k,j}, b_k\}$  przez element  $a_{k,k}$ ,
- 2) zerowanie pozostałych wartości  $k$ -tej kolumny przez odjęcie pierwszego równania pomnożonego przez współczynnik  $a_{i,k}$  od każdego z pozostałych równań.

Po  $N$  krokach algorytmu macierz  $\mathbf{A}$  jest przetworzona do macierzy jednostkowej, a wektor  $\mathbf{b}$  zawiera rozwiązanie układu równań, tj. poszukiwany wektor  $\mathbf{x}$ .

**Przykład:** Symbolicznie na przypadku  $2 \times 2$  ( $\mathbf{w}_i$  –  $i$ -ty wiersz,  $\mathbf{k}_j$  –  $j$ -ta kolumna)

Początkowa postać układu równań:

$$\begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

krok 1/1:  $\begin{bmatrix} 1 & \frac{a_{1,2}}{a_{1,1}} \\ a_{2,1} & a_{2,2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \frac{b_1}{a_{1,1}} \\ b_2 \end{bmatrix}$  (dzielimy  $\mathbf{w}_1$  przez element diagonalny  $a_{1,1}$ )

krok 1/2:  $\begin{bmatrix} 1 & \frac{a_{1,2}}{a_{1,1}} \\ 0 & \frac{a_{1,1}a_{2,2} - a_{2,1}a_{1,2}}{a_{1,1}} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \frac{b_1}{a_{1,1}} \\ \frac{a_{1,1}b_2 - a_{2,1}b_1}{a_{1,1}} \end{bmatrix}$  (zerujemy  $\mathbf{k}_1$ :  $\mathbf{w}_i \equiv \mathbf{w}_i - \mathbf{w}_1 \cdot a_{i,1}$ )

krok 2/1:  $\begin{bmatrix} 1 & \frac{a_{1,2}}{a_{1,1}} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \frac{b_1}{a_{1,1}} \\ \frac{a_{1,1}b_2 - a_{2,1}b_1}{a_{1,1}a_{2,2} - a_{2,1}a_{1,2}} \end{bmatrix}$  (dzielimy  $\mathbf{w}_2$  przez element diagonalny  $a_{2,2}$ )

krok 2/2:  $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \frac{a_{2,2}b_1 - a_{1,2}b_2}{a_{1,1}a_{2,2} - a_{2,1}a_{1,2}} \\ \frac{a_{1,1}b_2 - a_{2,1}b_1}{a_{1,1}a_{2,2} - a_{2,1}a_{1,2}} \end{bmatrix}$  (zerujemy  $\mathbf{k}_2$ :  $\mathbf{w}_i \equiv \mathbf{w}_i - \mathbf{w}_2 \cdot a_{i,2}$ )

Wynik zgadza się z uzyskanym poprzednio rozwiązaniem ze wzoru Cramera.

## Wybór elementu głównego (*pivoting*)

Problemem w dotychczas omówionym algorytmie jest możliwość wystąpienia wartości 0 na diagonali. Aby uniknąć dzielenia przez zerowy element diagonalny dokonuje się wyboru elementu głównego przez:

- zmianę kolejności wierszy-równań (*partial pivoting*)
- zmianę kolejności wierszy-równań i kolumn-zmiennych (*full pivoting*) co wymaga odtworzenia pierwotnej kolejności zmiennych po rozwiązaniu układu.

Algorytm eliminacji Gaussa-Jordana z wyborem jako elementu głównego największego elementu w kolumnie/wierszu jest dodatkowo bardziej odporny na błędy reprezentacji danych.

## Eliminacja Gaussa do macierzy trójkątnej i rozwiązywanie metodą podstawienia wstecz

Proces eliminacji Gaussa-Jordana jest prowadzony aż do uzyskania macierzy jednostkowej z lewej strony równania i rozwiązywania po prawej stronie równania. Taki sposób rozwiązywania układu równań może być uproszczony przez ograniczenie drugiej operacji algorytmu (zerowanie kolumny) do elementów znajdujących się pod diagonalą. W wyniku tej modyfikacji proces eliminacji przekształca A do macierzy trójkątnej górnej (zamiast macierzy diagonalnej). Rozwiązywanie takiego układu jest łatwe do zaprogramowania metodami iteracyjnymi. Taka odmiana metody eliminacji nosi nazwę eliminacji Gaussa (*Gaussian elimination*).

**Rozwiązywanie układu trójkątnego górnego przez podstawienie wstecz (back substitution):**

$$\left[ \begin{array}{cccc|c} u_{1,1} & \cdots & \cdots & u_{1,N} & x_1 \\ 0 & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & u_{N,N} & x_N \end{array} \right] = \left[ \begin{array}{c} b_1 \\ \vdots \\ \vdots \\ b_N \end{array} \right]$$

↑       $x_N = ?$

Dla układu równań w postaci:

Rozwiązywanie iteracyjne ma postać:

Począwszy od  $x_N = \frac{b_N}{u_{N,N}}$  każdy kolejny:  $x_k = \frac{b_k - \sum_{j=k+1}^N u_{k,j}x_j}{u_{k,k}}, \quad k = N-1, \dots, 1$

**Przykład:** sprawdźmy zgodność rozwiązania przypadku  $2 \times 2$  z wynikami innych metod.

Początkowa postać układu równań:

$$\begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \quad \mathbf{x}=?$$

krok E/2:

$$\begin{bmatrix} a_{1,1} & a_{1,2} \\ 0 & \frac{a_{1,1}a_{2,2}-a_{1,2}a_{2,1}}{a_{1,1}} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ \frac{a_{1,1}b_2-a_{2,1}b_1}{a_{1,1}} \end{bmatrix} \quad (\text{zerujemy elementy pod diagonalą } \mathbf{w}_i = \mathbf{w}_i - \mathbf{w}_1 \cdot a_{i,1}/a_{1,1})$$

Dalej **rozwiązujemy iterując wstecz:**

krok I/2:  $x_2 = \frac{a_{1,1}b_2-a_{2,1}b_1}{a_{1,1}a_{2,2}-a_{1,2}a_{2,1}}$

krok I/1:  $x_1 = \frac{b_1-x_2a_{1,2}}{a_{1,1}} = \frac{a_{2,2}b_1-a_{1,2}b_2}{a_{1,1}a_{2,2}-a_{1,2}a_{2,1}}$

Wynik się zgadza z wynikami poprzednich metod.

## Operacje eliminacji Gaussa w sensie macierzowym

W każdym kroku eliminacji Gaussa są wykonywane operacje zerujące elementy kolumny macierzy leżące pod elementem diagonalnym. Można to zapisać operacją mnożenia przez macierz modyfikującą:

$$\mathbf{A}^{(i)} = \mathbf{M}^{(i)} \mathbf{A}^{(i-1)} = \begin{bmatrix} 1 & 0 & & \cdots & 0 \\ 0 & \ddots & \ddots & & \vdots \\ \vdots & 0 & 1 & 0 & \vdots \\ 0 & 0 & \frac{-a_{i+1,i}}{a_{i,i}} & 1 & 0 \\ \vdots & \vdots & \vdots & 0 & 0 \\ 0 & 0 & \frac{-a_{N,i}}{a_{i,i}} & 0 & 0 \end{bmatrix} \begin{bmatrix} a_{1,1} & \cdots & a_{1,i} & \cdots & a_{1,N} \\ 0 & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & a_{i,i} & \vdots \\ \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & a_{N,i} & \cdots & a_{N,N} \end{bmatrix}$$

Cały proces eliminacji Gaussa prowadzi do macierzy trójkątnej górnej więc możemy napisać:

$$\boxed{\mathbf{M}^{(N-1)} \cdots \mathbf{M}^{(2)} \mathbf{M}^{(1)} \mathbf{A} = \mathbf{U}}$$

Przekształcając, dostajemy:

$$\mathbf{A} = \boxed{(\mathbf{M}^{(N-1)} \cdots \mathbf{M}^{(2)} \mathbf{M}^{(1)})^{-1} \mathbf{U}} = \mathbf{L} \mathbf{U}$$

Macierz w nawiasie jest (tak samo jak jej odwrotność  $\mathbf{L}$ ) trójkątna dolna z jedynkami na diagonali.

## Metoda dekompozycji LU i jej zastosowania

Poważną wadą poprzednio przedstawionych metod eliminacji jest operowanie jednocześnie na macierzy **A** i na wektorze **b**. W przypadku powtarzalnych układów równań ze zmienną lewą stroną (jak np. w metodach różnic/elementów skończonych ze zmiennymi warunkami brzegowymi) oznacza to konieczność wykonania wszystkich operacji powtórnie. Przedstawiana poniżej metoda dekompozycji **LU** polega w istocie na zapisaniu wykonywanych operacji modyfikujących macierz **A** i wektor **b** w postaci dwóch oddzielnych macierzy transformacji. W ten sposób po zmianie danych **b** wystarczy (przy niezmienionej macierzy **A**) wykonać na nich operacje z macierzy transformacji. Z użyciem macierzy transformacji łatwo również wyznaczyć dla **A** wyznacznik i macierz odwrotną. Z tego powodu jest to podstawowa metoda rozwiązywania układów równań. W metodzie dekompozycji LU macierz A jest rozdzielana na iloczyn dwu macierzy składowych - dolnej i górnej trójkątnej, tzn.  $\mathbf{A} = \mathbf{L}\mathbf{U}$ . Dla przykładu macierz **A** o wymiarach (3,3) jest

dekomponowana na składowe:

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{vmatrix} \cdot \begin{vmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{vmatrix}$$

Sama zasada dekompozycji (nazywana algorytmem Doolittle'a) opiera się na efekcie przyrostu o jedną nową zmienną w każdym kolejnym elemencie wiersza/kolumny:

$$u_{i,j} = a_{i,j} - \sum_{k=1}^{i-1} l_{i,k} u_{k,j}, \quad j \geq i$$

$$l_{i,j} = \frac{1}{u_{j,j}} \left( a_{i,j} - \sum_{k=1}^{j-1} l_{i,k} u_{k,j} \right), \quad j < i$$

**Przykład:** Analiza implementacji algorytmu Doolittle'a

```
[N, M]= size(A);  
L=eye(N);  
U=zeros(N);  
for i=1:N  
    for j=i:N  
        U(i,j)=A(i,j);  
        for k=1:(i-1)  
            U(i,j)=U(i,j)-L(i,k)*U(k,j);  
        end  
    end  
    for j=i+1:N  
        L(j,i)=A(j,i);  
        for k=1:(i-1);  
            L(j,i)=L(j,i)-L(j,k)*U(k,i);  
        end  
        L(j,i)=L(j,i)/U(i,i);  
    end  
end
```

## Rozwiązywanie LU-zdekomponowanego układu równań liniowych

Dysponując rozkładem macierzy A na macierze trójkątne górną i dolną, możemy zastosować znany już iteracyjny schemat przez podstawienie wstecz i dodatkowo przez podstawienie w przód.

Ponieważ:

$$\mathbf{A} \cdot \mathbf{x} = (\mathbf{L} \cdot \mathbf{U}) \cdot \mathbf{x} = \mathbf{L} \cdot (\mathbf{U} \cdot \mathbf{x}) = \mathbf{b}$$

to rozwiązuje najpierw:  $\mathbf{L} \cdot \mathbf{y} = \mathbf{b}$  przez podstawienie w przód:

$$y_1 = b_1, \quad y_i = b_i - \sum_{j=1}^{i-1} l_{i,j} y_j, \quad i = 2, 3, \dots, N$$

a następnie:

$\mathbf{Ux} = \mathbf{y}$  przez podstawienie wstecz:

$$x_N = \frac{y_N}{u_{N,N}}, \quad x_i = \frac{1}{u_{i,i}} \left( y_i - \sum_{j=i+1}^N u_{i,j} x_j \right), \quad i = N-1, N-2, \dots, 1$$

## Inne zastosowania macierzy dekompozycji LU

**Obliczenie wyznacznika:** wyznacznik to prosty iloczyn elementów diagonalnych U

$$\det[\mathbf{A}] = \det[\mathbf{L}] \cdot \det[\mathbf{U}] = \prod_{j=1}^N u_{j,j}$$

**Wyznaczenie macierzy odwrotnej  $\mathbf{A}^{-1}$ :** Ponieważ iloczyn macierzy i macierzy odwrotnej jest macierzą jednostkową ( $\mathbf{LU}\mathbf{A}^{-1} = \mathbf{I}$ ), to odwrotność macierzy możemy łatwo wyznaczyć wykonując podstawienia w przód/wstecz na kolumnach macierzy jednostkowej, tj. rozwiązuając:

$$\mathbf{L}(\mathbf{U}\mathbf{a}_k^{-1}) = [0, \dots, 0, 1_k, 0, \dots, 0]^T, \quad \mathbf{a}_k^{-1} - k\text{-ta kolumna macierzy } \mathbf{A}^{-1}$$

## Szczególny przypadek dekompozycji LU - dekompozycja Choleskiego

Jeśli macierz  $\mathbf{A}$  układu równań jest macierzą symetryczną dodatnio określona (jest np. macierzą kowariancyjną lub korelacyjną), to jej dekompozycja LU ma prostszą postać nazywaną dekompozycją Choleskiego – macierz trójkątna górną  $\mathbf{U}$  ma taką samą zawartość elementową jak macierz trójkątna dolna  $\mathbf{L}$ . Wyznaczyć trzeba dwukrotnie mniej elementów macierzy.

Dekompozycja ma postać  $\mathbf{A} = \begin{bmatrix} a_{1,1} & \dots & \dots & a_{1,N} \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ a_{N,1} & \dots & \dots & a_{N,N} \end{bmatrix} = \mathbf{LL}^T$ ,  $\mathbf{L} = \begin{bmatrix} l_{1,1} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ l_{N,1} & \dots & \dots & l_{N,N} \end{bmatrix}$

Poszczególne elementy macierzy  $\mathbf{L}$  są wyznaczane wg zależności:

$$l_{1,1} = \sqrt{a_{1,1}}, \quad l_{i,i} = \sqrt{a_{i,i} - \sum_{k=1}^{i-1} l_{i,k}^2}, \quad i = 2, 3, \dots, N \quad l_{i,j} = \frac{a_{i,j} - \sum_{k=1}^{j-1} l_{i,k} l_{j,k}}{l_{j,j}}, \quad j = 2, 3, \dots, i-1$$

**Przykład:**

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 8 & 10 \\ 3 & 10 & 22 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 2 & 0 \\ 3 & 2 & 3 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \\ 0 & 2 & 2 \\ 0 & 0 & 3 \end{bmatrix} = \mathbf{LL}^T$$

### Przykład: Równanie Yule'a-Walkera

Klasyka cyfrowego przetwarzania sygnałów to równanie Yule'a-Walkera, opisujące m.in. model autoregresyjny (model AR) sygnału używany w kompresji mowy LPC (np. w telefonii komórkowej) oraz przejście sygnału losowego przez filtr (kanał), opisany odpowiedzią impulsową (model MA). Równanie to jest również uogólnieniem estymatora LS na przypadek nieskończenie długich sygnałów wejściowych o znanych parametrach dynamicznych.

Rozpatrzmy zadanie wyznaczania odpowiedzi impulsowej filtra na podstawie długotrwałych rejestracji jego wejścia  $u$  i wyjścia  $y$ . Podstawą równań jest zależność splotowa obiektu, opisanego odpowiedzią impulsową  $h$ :

$$y_i = \sum_{k=0}^{N-1} h_k u_{i-k} \longrightarrow r_i^{uy} = E[u_i y_i] = \sum_{k=0}^{N-1} h_k E[u_i u_{i-k}] = \sum_{k=0}^{N-1} h_k r_{i-k}^{uu}$$

Drugą zależność, łączącą korelacje własną  $r^{uu}$  i wzajemną  $r^{uy}$  sygnałów obiektu, uzyskujemy mnożąc obie strony równania przez  $u_i$  i biorąc ich wartość oczekiwana. Zapisując tę zależność dla kolejnych chwil czasowych uzyskujemy układ  $N$  równań z poszukiwanymi współczynnikami filtra:

$$\begin{bmatrix} r_0^{uu} & r_1^{uu} & \dots & r_{N-1}^{uu} \\ r_1^{uu} & r_0^{uu} & \ddots & r_1^{uu} \\ \vdots & \ddots & \ddots & \vdots \\ r_{N-1}^{uu} & \dots & r_1^{uu} & r_0^{uu} \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_{N-1} \end{bmatrix} = \begin{bmatrix} r_0^{uy} \\ r_1^{uy} \\ \vdots \\ r_{N-1}^{uy} \end{bmatrix}$$

Macierz po lewej stronie jest symetryczna i dodatnio określona. Układ może być efektywnie rozwiązywany przy długiej odpowiedzi  $h$  metodą dekompozycji Choleskiego i podstawienia.

**Przykład:** Równanie Yule'a-Walkera dla bezprzewodowego kanału komunikacyjnego z odbiciami i sygnału referencyjnego PRBS

Sygnal PRBS jest wysyłany przez nadajnik i jest znany odbiornikowi. Odbiornik koreluje odebrany sygnał ze znaną sekwencją PRBS uzyskując wartości korelacji wzajemnej  $r$ . Jeśli kanał bezprzewodowy ma krótką odpowiedź impulsową, to wystarczy rozwiązać mały układ równań uformowany na bazie długiej sekwencji metodami statystyki (korelacja).

Na przykład, dla sekwencji PRBS o długości  $N=32767$  znane są wartości autokorelacji. Na diagonali macierzy autokorelacji znajduje się wartość  $m$  i wartość  $c$  poza nią, gdzie  $m$  jest ilością bitów niezerowych, a  $c=m(m-1)/(N-1)$ .

Zakładając, że kanał komunikacyjny ma odpowiedź impulsową nie dłuższą niż 4 próbki (czas trwania czterech symboli transmisji) układamy równanie:

$$\begin{bmatrix} m & c & c & c \\ c & m & c & c \\ c & c & m & c \\ c & c & c & m \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix} = \begin{bmatrix} r_0 \\ r_1 \\ r_2 \\ r_3 \end{bmatrix} \quad h=?$$

Rozwiązuje się je ze względu na  $h$  i uzyskujemy oszacowanie odpowiedzi impulsowej kanału. Porównaj z zapisem splotowym wiążącym wejście (sygnał PRBS) z wyjściem kanału (odebrany sygnał) jak przedstawiono na poprzednim wykładzie. Czy, mimo dużo mniejszego rozmiaru macierzy, wykorzystujemy tyle samo informacji ?

## Rozwiązywanie układu równań liniowych w Matlabie

Narzucającym się rozwiążaniem układu równań  $\mathbf{Ax} = \mathbf{b}$  jest mnożenie prawej strony przez odwrotność  $\mathbf{A}$ , tzn.  $\mathbf{x} = \text{inv}(\mathbf{A})\mathbf{b}$ . Nie jest to najlepsze możliwe rozwiązanie, ponieważ omówione dotąd zasady eliminacji i podstawiania (*direct solution*) są zaimplementowane w Matlabie przy użyciu lewostronnego dzielenia macierzowego w postaci  $\mathbf{x} = \mathbf{A} \setminus \mathbf{b}$  i dopasowywane do aktualnej postaci macierzy  $\mathbf{A}$ . Rozróżniane są m.in. ogólne macierze kwadratowe (dekompozycja LU i podstawienie), trójkątne dolna i górna (tylko podstawienie), trójprzekątniowa (szybka eliminacja bez pivotingu), rzadka (szybka eliminacja), symetryczna dodatnio określona (dekompozycja Choleskiego).

Metoda *direct solution* jest **2 do 3 razy szybsza** od metody macierzy odwrotnej, daje również **dokładniejsze wyniki** (na poziomie błędu reprezentacji).

Funkcja odwracania macierzy również wykorzystuje dekompozycję LU, ale modyfikacja prawej strony równania jest osobnym krokiem obliczeń.

## Metody dla macierzy specjalnych i rzadkich

Duże macierze **rzadkie** (częsty przypadek przy analizie metodą różnic i elementów skończonych) i w szczególności macierze **trójprzekątniowe** (interpolacja spline'ami) mogą być przetwarzane metodami eliminacji dużo efektywniej przy wykrywaniu i pomijaniu elementów zerowych. Macierze rzadkie są często pamiętane w strukturach upakowanych (nie w pełnej macierzy) co przy tak prowadzonej eliminacji daje duże oszczędności pamięci.

## Metody iteracyjne – alternatywa dla macierzy rzadkich

Wadą metod bezpośrednich wykorzystujących eliminację i podstawienie jest mała efektywność dla macierzy rzadkich. Dekompozycja ogólnej macierzy rzadkiej (o niewielu elementach poza diagonalą) może prowadzić do pełnej macierzy trójkątnej po dekompozycji. Alternatywą dla metod bezpośrednich są metody iteracyjne, w których modyfikacji nie podlega sama macierz równania a jedynie kolejne oszacowania rozwiązania, bazujące na niezerowych elementach macierzy.

Ideę metod iteracyjnych tłumaczy poniższy przykład:

$$\left| \begin{array}{ccc|c} a_{1,1} & a_{1,2} & a_{1,3} & x_1 \\ a_{2,1} & a_{2,2} & a_{2,3} & x_2 \\ a_{3,1} & a_{3,2} & a_{3,3} & x_3 \end{array} \right| = \left| \begin{array}{c} b_1 \\ b_2 \\ b_3 \end{array} \right|$$

Ogólny układ równań:

możemy zapisać w równoważnej postaci trzech równań:

$$x_1 = \frac{b_1 - (a_{1,2}x_2 + a_{1,3}x_3)}{a_{1,1}}, \quad x_2 = \frac{b_2 - (a_{2,1}x_1 + a_{2,3}x_3)}{a_{2,2}}, \quad x_3 = \frac{b_3 - (a_{3,1}x_1 + a_{3,2}x_2)}{a_{3,3}}$$

Przyjmując, że powyższa zależność ma charakter iteracyjny, tzn. po lewej stronie są nowe oszacowania  $x$  oparte na poprzednich wartościach  $x$ , możemy napisać zależność iteracyjną:

$$x_1^{(k+1)} = \frac{b_1 - (a_{1,2}x_2^{(k)} + a_{1,3}x_3^{(k)})}{a_{1,1}}, \quad x_2^{(k+1)} = \frac{b_2 - (a_{2,1}x_1^{(k)} + a_{2,3}x_3^{(k)})}{a_{2,2}}, \quad x_3^{(k+1)} = \frac{b_3 - (a_{3,1}x_1^{(k)} + a_{3,2}x_2^{(k)})}{a_{3,3}}$$

i wyliczać nowe oszacowania rozwiązania począwszy od wybranej wartości startowej.

Zapisując ostatnią zależność w postaci macierzowej otrzymujemy iterację metodą Jacobiego:

$$\mathbf{x}^{(k+1)} = \mathbf{D}^{-1} \left( \mathbf{b} - (\mathbf{L} + \mathbf{U}) \mathbf{x}^{(k)} \right)$$

gdzie:  $\mathbf{D}$  – jest diagonalą macierzy  $\mathbf{A}$ ,

$\mathbf{L}$  – jest dolną trójkątną częścią (nie dekompozycją LU !)  $\mathbf{A}$ ,

$\mathbf{U}$  – jest górną trójkątną częścią  $\mathbf{A}$ .

**Metoda Gaussa-Seidla** jest prostą modyfikacją tej metody wykorzystującą fakt, że w momencie wyliczania elementu  $x_j$  są już znane świeże (bardziej aktualne) oszacowania rozwiązania dla elementów o niższych indeksach. Taka modyfikacja daje szybszą zbieżność metody i ma postać:

$$\mathbf{x}^{(k+1)} = \mathbf{D}^{-1} \left( \mathbf{b} - (\mathbf{L} \mathbf{x}^{(k+1)} + \mathbf{U} \mathbf{x}^{(k)}) \right)$$

**Metoda Successive Overrelaxation (SOR)** została opracowana w celu osiągnięcia szybszej zbieżności schematu iteracyjnego do rozwiązania. W każdym kroku iteracyjnym rozwiązanie jest ważoną średnią poprzedniego rozwiązania i nowego oszacowania:

$$\mathbf{x}^{(k+1)} = \omega \mathbf{D}^{-1} \left( \mathbf{b} - (\mathbf{L} \mathbf{x}^{(k+1)} + \mathbf{U} \mathbf{x}^{(k)}) \right) + (1 - \omega) \mathbf{x}^{(k)}$$

Optymalny dobór wagi  $\omega$  nazywanej współczynnikiem relaksacji (*relaxation parameter*) zależy od własności macierzy  $\mathbf{A}$ . Najszybszą zbieżność daje w średnim przypadku wartość 1.2 – 1.3.

Analiza zbieżności poszczególnych metod iteracyjnych wykracza poza przyjęty zakres wykładu.

## Podsumowanie, kluczowe elementy tego wykładu:

- „Szkolne” metody rozwiązywania układów równań (metoda wyznaczników, metoda podstawień) nie są stosowane w obliczeniach komputerowych ze względu na złożoność obliczeniową
- Metodą stosowaną współcześnie dla ogólnej klasy macierzy jest **dekompozycja LU**
- Szczególna struktura macierzy może być wykorzystana do zwiększenia szybkości obliczeń lub zmniejszenia zużycia pamięci podczas obliczeń
  - Równania z macierzami symetrycznymi i dodatnio określonymi (macierz kowariancji, korelacji) mogą być rozwiązyane szybciej przez **dekompozycję Cholesky'ego** ( $L=U^T$ )
  - Równania z macierzami rzadkimi (mało elementów niezerowych) o dużych rozmiarach można rozwiązać **metodami iteracyjnymi** przy mniejszym zużyciu pamięci
- Rozwiązywanie równania macierzowego w Matlabie można wyznaczyć przez `y=inv(A)*x`, ale szybciej i dokładniej będzie przez `y=A\x`. Dlaczego ?