

POLITECHNIKA ŁÓDZKA

WYDZIAŁ FIZYKI TECHNICZNEJ, INFORMATYKI I
MATEMATYKI STOSOWANEJ

Kierunek: Matematyka

Specjalność: Matematyczne Metody Analizy Danych Biznesowych

Algorytmy dekompozycji QR

Anna Szczepaniak
Nr albumu: 210094

Praca licencjacka
napisana w Instytucie Matematyki Politechniki Łódzkiej

Promotor: dr inż. Piotr Kowalski

ŁÓDŹ, WRZESIEŃ 2019

Spis treści

1	Wstęp	2
2	Preliminaria	3
2.1	Oznaczenia użyte w pracy	3
2.2	Elementy algebry liniowej	3
3	Algorytm QR	7
3.1	Algorytm QR metodą odbić Householdera	10
3.2	Algorytm QR metodą rotacji Givensa	12
4	Eksperymenty numeryczne	17
4.1	Algorytm dekompozycji QR metodą odbić Householdera	17
4.2	Algorytm dekompozycji QR metodą rotacji Givensa	18
4.3	Eksperymenty numeryczne - porównanie jakości dwóch algorytmów	19
4.3.1	Test dla macierzy małych rozmiarów	19
4.3.2	Test dla macierzy średniego rozmiaru	20
4.3.3	Test dla macierzy dużych rozmiarów	21
4.3.4	Podsumowanie	22
5	Podsumowanie	24

Rozdział 1

Wstęp

Tematem niniejszej pracy jest algorytm QR dekompozycji macierzy, który to został wynaleziony w 1961 roku przez Francisa i Kubłanowską. Jest on jedną z efektywniejszych znanych metod rozwiązywania pełnego zadania własnego dla macierzy symetrycznych lub niesymetrycznych. Należy on do najważniejszych algorytmów opracowanych w XX wieku. Pomimo tego, iż w czasie studiów podobne operacje wykonywane są za pomocą procesu ortogonalizacji Grama-Schmidta, sam algorytm dekompozycji QR rozwiązywany jest na inne sposoby, które dalej zostaną dokładnie omówione. W pracy tej chcieliśmy zgromadzić informacje o samym algorytmie, matematycznych podstawach na których jest oparty oraz matematycznie opisać operacje, które są wykonywane przez rzeczywiste implementacje.

Praca uporządkowana jest w następujący sposób. W rozdziale 2 znajdują się preliminaria, w których umieszczone zostały wybrane elementy algebry liniowej wraz z kilkoma znanymi lematami, które zostały udowodnione samodzielnie na potrzeby tej pracy. W rozdziale 3 znajduje się twierdzenie o rozkładzie QR wraz z dowodem. Dowód ten został samodzielnie opracowany na podstawie szkiców z książek [3, 4, 5, 6]. W tym samym rozdziale omówimy również podstawy działania algorytmów służących do dekompozycji, tj. algorytmu QR metodą odbić Householdera oraz algorytmu QR metodą rotacji Givensa. W rozdziale 4 przedstawiamy opracowane własne implementacje z wykorzystaniem dokumentacji języka R [1] wraz z eksperymentami numerycznymi oraz obserwacjami jakie poczyniliśmy przy tychże eksperymentach. W rozdziale 5 zaprezentowane są wnioski z wyników naszej pracy w rozdziale 4.

Rozdział 2

Preliminaria

W niniejszym rozdziale przypomniane zostaną wybrane elementy z zakresu algebry liniowej, potrzebne do wyjaśnienia działania algorytmu QR.

2.1 Oznaczenia użyte w pracy

Poniżej prezentuję oznaczenia jakich będziemy używać w pracy:

- $\mathbb{R}^{m \times n}$ - przestrzeń wszystkich macierzy o m wierszach i n kolumnach
- span - przestrzeń rozpięta na wektorach
- $(\cdot|\cdot)$ - iloczyn skalarny
- $\|\cdot\|$ - norma
- A, B - macierze
- x - szukany wektor
- \mathbf{b} - wektor prawej strony równania
- I - macierz identycznościowa

2.2 Elementy algebry liniowej

Niniejszą część rozpoczniemy od zdefiniowania pojęcia macierzy. Warto nadmienić, że w znaczącej części literatury - pojęcie to nie jest definiowane w sposób matematycznie precyzyjny.

Definicja 2.1 (Macierz [5, Def. 8.1]).

Niech $n, m \in \mathbb{N}$. Macierzą o m wierszach oraz n kolumnach (nazywaną również macierzą o wymiarach m na n) i wyrazach w ciele \mathbb{R} nazywamy funkcję

$$A : \{1, 2, \dots, m\} \times \{1, 2, \dots, n\} \rightarrow \mathbb{R}.$$

Wartością funkcji dla argumentu (i, j) , gdzie $i \in \{1, 2, \dots, m\}$, $j \in \{1, 2, \dots, n\}$ jest element $a_{ij} \in \mathbb{R}$. Macierz często zapisujemy w postaci tabeli

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}.$$

Przez $\mathbb{R}^{m \times n}$ oznaczmy zbiór wszystkich macierzy o wymiarach m na n i elementach z \mathbb{R} .

Macierze posiadają wiele istotnych dla nas podklas, posiadających określone cechy. Wymienimy kilka typów macierzy, szczególnie interesujących z perspektywy omawianego przez nas tematu.

Definicja 2.2 (Rodzaje macierzy [5, Sekcja 8.1]).

Możemy rozróżnić następujące rodzaje macierzy:

- *Macierz kwadratową o wymiarze n na n nazywamy macierz o równej liczbie wierszy i kolumn. Liczbę n nazywamy wtedy stopniem macierzy kwadratowej.*
- *Jeśli w macierzy kwadratowej $[a_{ij}]$ wszystkie elementy poza główną przekątną są równe zero, to taką macierz nazywamy macierzą diagonalną, oznaczamy ją jako $\text{diag}(a_{11}, a_{22}, \dots, a_{nn})$.*
- *Macierz jednostkową stopnia n nazywamy taką macierz diagonalną, w której wszystkie elementy na głównej przekątnej są równe 1.*
- *Macierz symetryczną nazywamy macierz kwadratową $A = [a_{ij}]$, której wyrazy spełniają warunek*

$$\forall_{i,j \in \{1, \dots, n\}} \quad a_{ij} = a_{ji}.$$

- *Jeśli w macierzy kwadratowej $[a_{ij}]$ wszystkie elementy poniżej głównej przekątnej są równe 0, to taką macierz nazywamy macierzą trójkątną górną. Analogicznie jeśli w macierzy kwadratowej $[a_{ij}]$ wszystkie elementy powyżej głównej przekątnej są równe 0 to taką macierz nazywamy trójkątną dolną.*
- *Niech $A = [a_{ij}] \in \mathbb{R}^{m \times n}$. Macierz $D \in \mathbb{R}^{n \times m}$ nazywamy macierzą transponowaną do A , jeśli*

$$d_{ij} = a_{ji},$$

dla dowolnych $i \in \{1, \dots, n\}$ oraz $j \in \{1, \dots, m\}$. Macierz D oznaczamy symbolem A^T .

Następujące lematy zostały opracowane na potrzeby dowodu z rozdziału 3.

Lemat 2.3 (O iloczynie macierzy trójkątnych górnych).

Niech $A, B \in \mathbb{R}^{n \times n}$ będą dwiema nieosobliwymi macierzami trójkątnymi górnymi. Wtedy

$$C = A \cdot B,$$

jest też nieosobliwą macierzą trójkątną górną. Ponadto, jeśli macierze A i B mają na przekątnej same 1, to również macierz C tak ma.

Dowód. Niech A, B jak w założeniach, oraz $C = A \cdot B$. Korzystając z twierdzenia Cauchy'ego o iloczynie macierzy otrzymujemy natychmiast, że macierz $C \in \mathbb{R}^{n \times n}$ jest również nieosobliwa. Pozostaje pokazać, że jest macierzą trójkątną górną. Rozważmy dowolną $i, j \in \{1, \dots, n\}$ parę indeksów poniżej przekątnej, tj. takich dla których $i > j$. Wtedy

$$c_{ij} = a_{i1}b_{1j} + \dots + a_{in}b_{nj}. \quad (2.1)$$

Zauważmy, że elementy $a_{i1}, \dots, a_{i(i-1)}$ są równe 0 gdyż pochodzą z macierzy trójkątnej górnej. Ponadto elementy $b_{j(j+1)}, \dots, b_{jn}$ są z analogicznego powodu również równe 0. Skoro $i > j$ to każdy składnik sumy (2.1) posiada czynnik równy 0. Zatem c_{ij} dla rozważanego indeksu jest równe 0. W dowolności wyboru i, j macierz C jest macierzą trójkątną górną. Załóżmy dalej, że A i B mają na przekątnej same 1. Niech $i \in \{1, \dots, n\}$. Wtedy

$$c_{ii} = a_{i1}b_{1i} + \dots + a_{in}b_{ni} = \sum_{k=1}^n a_{ik}b_{ki} = a_{ii}b_{ii} = 1.$$

Z dowolności wyboru i macierz C ma na przekątnej same 1. □

Lemat 2.4 (O odwracaniu macierzy trójkątnej górnej).

Niech $A \in \mathbb{R}^{n \times n}$ będzie nieosobliwą macierzą trójkątną górną. Wtedy istnieje macierz $A^{-1} = [\bar{a}_{ij}]_{i=1, \dots, n}^{j=1, \dots, n}$ i jest ona nieosobliwą macierzą trójkątną górną. Ponadto jeśli macierz A ma na przekątnej wyłącznie 1, to również macierz A^{-1} ma na przekątnej same 1.

Dowód. Niech A będzie macierzą jak w założeniach twierdzenia. Na mocy twierdzenia 9.23 z [5] i twierdzenia 9.22 z [5], i nieosobliwości macierzy A , wiemy że macierz A^{-1} istnieje i jest nieosobliwa. Pozostaje pokazać, że jest macierzą trójkątną górną. Pokazać ten fakt można na kilka różnych sposobów, zaprezentujemy tutaj najprostszy pod względem wiedzy teoretycznej dowód. Wykorzystamy drugi krok z algorytmu eliminacji Gaussa. Przypomnijmy, że algorytm ten służy rozwiązaniu układu równań postaci

$$A\mathbf{x} = \mathbf{b},$$

i składa się z dwóch kroków. Pierwszym jest redukcja macierzy metodą eliminacji do macierzy trójkątnej górnej. Drugim etapem jest iteracyjne policzenie rozwiązania. Przyjmijmy, że mamy poniższy układ równań z macierzą A :

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ 0 & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}.$$

Wtedy również $\mathbf{x} = A^{-1}\mathbf{b}$, gdzie $A^{-1} = [\bar{a}_{ij}]$. Skoro macierz A jest nieosobliwa to wiemy, że elementy na przekątnej są różne od 0, tzn.

$$\forall k \in \{1, \dots, n\} \quad a_{kk} \neq 0.$$

Wtedy prawdziwy jest wzór pozwalający na iteracyjne policzenie wektora \mathbf{x} , tj. dla każdego k począwszy od n do 1.

$$x_k = \frac{1}{a_{kk}} \left(b_k - \sum_{i=k+1}^n a_{ki} x_i \right). \quad (2.2)$$

Przypuśćmy nie wprost, że macierz A^{-1} nie jest trójkątną górną. Istnieje zatem takie $i, j \in \{1, \dots, n\}$, $i > j$ i $\bar{a}_{ij} \neq 0$. Rozważmy $\mathbf{b} = [0, 0, \dots, 1, 0, 0, \dots]^T$, gdzie $\mathbf{b}_j = 1$. Wtedy dla każdego $k > i > j$, $k \in \{1, \dots, n\}$, $x_k = 0$ oraz $x_i = \frac{1}{a_{ii}} \cdot (0 - 0) = 0$. Z drugiej strony $\mathbf{x} = A^{-1}\mathbf{b}$. Stąd

$$x_i = \bar{a}_{i1}\mathbf{b}_1 + \bar{a}_{i2}\mathbf{b}_2 + \dots + \bar{a}_{in}\mathbf{b}_n = \bar{a}_{ij}\mathbf{b}_j = \bar{a}_{ij} \neq 0.$$

Sprzeczność jest efektem przypuszczenia, że macierz A^{-1} nie jest macierzą trójkątną górną. Rozważmy dalej przypadek, że macierz A ma na przekątnej same 1. Przypuśćmy nie wprost, że macierz A^{-1} nie ma wyłączenie 1 na przekątnej. Istnieje zatem takie $i \in \{1, \dots, n\}$, że $\bar{a}_{ii} \neq 1$. Rozważmy $\mathbf{b} = [0, 0, \dots, 1, 0, 0, \dots]^T$, gdzie $\mathbf{b}_i = 1$. Wtedy dla każdego $k > i$, $\mathbf{x}_k = 0$ oraz $x_i = \frac{1}{a_{ii}} \cdot (1 - 0) = 1$. Z drugiej strony $\mathbf{x} = A^{-1}\mathbf{b}$. Stąd

$$x_i = \bar{a}_{i1}\mathbf{b}_1 + \bar{a}_{i2}\mathbf{b}_2 + \dots + \bar{a}_{in}\mathbf{b}_n = \bar{a}_{ii}\mathbf{b}_i = \bar{a}_{ii} \neq 1.$$

Sprzeczność jest efektem przypuszczenia, że macierz A^{-1} nie ma na przekątnej wyłącznie 1. □

Definicja 2.5 (Iloczyn macierzy [5, Def. 9.13]).

Niech $A \in \mathbb{R}^{m \times n}$ i $B \in \mathbb{R}^{m \times n}$. Jeśli

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}, B = \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1m} \\ b_{21} & b_{22} & \dots & b_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{k1} & b_{k2} & \dots & b_{km} \end{bmatrix},$$

to iloczynem macierzy B i A nazywamy taką macierz $C = [c_{lj}]_{j=1,\dots,n}^{l=1,\dots,k}$, że dla $i = 1, \dots, m, j = 1, \dots, n$

$$c_{lj} = \sum_{l=1}^m b_{li} \cdot a_{ij}.$$

Definicja 2.6 (Macierz odwrotna [5, Def. 9.21]).

Macierz A jest odwracalna, jeśli istnieje taka macierz B , że zachodzi

$$A \cdot B = B \cdot A = I,$$

gdzie I jest macierzą jednostkową. Macierz B nazywa się wówczas macierzą odwrotną do macierzy A i oznacza się przez A^{-1} . Macierz A nazywamy nieosobliwą, jeśli istnieje macierz B , która jest do niej odwrotna.

Definicja 2.7 (Macierz ortogonalna [5, Def. 14.26]).

Macierz kwadratową $A \in \mathbb{R}^{n \times n}$ nazywamy macierzą ortogonalną, jeśli spełniony jest następujący warunek:

$$A^T \cdot A = A \cdot A^T = I_n.$$

Lemat 2.8 (O iloczynie macierzy ortogonalnych).

Niech $A, B \in \mathbb{R}^{n \times n}$ będą dwiema ortogonalnymi macierzami. Wówczas

$$C = A \cdot B$$

jest też macierzą ortogonalną.

Dowód. Niech A i B będą dwiema ortogonalnymi macierzami, tzn. że zachodzi:

$$A^T A = I,$$

$$B^T B = I.$$

Niech dalej $C = AB$. Należy sprawdzić, czy $C^T C = I$. Istotnie

$$C^T C = B^T A^T A B = B^T I B = B^T B = I.$$

Zatem C jest ortogonalna. □

Definicja 2.9 (Wektor [5, Sekcja 8.1]).

Wektorem (kolumnowym) nazywamy macierz z przestrzeni $\mathbb{R}^{n \times 1}$.

Tak zdefiniowane wektory można utożsamiać z elementami przestrzeni \mathbb{R}^n .

Definicja 2.10 (Liniowa niezależność wektorów [5, Def. 7.21 oraz Def. 7.22]).

Niech $\mathbf{v}_1, \dots, \mathbf{v}_n$ będą różnymi elementami przestrzeni liniowej \mathbb{R}^k . Zbiór wektorów $\mathbf{v}_1, \dots, \mathbf{v}_n$ nazywamy liniowo niezależnymi jeżeli

$$\forall_{a_1, \dots, a_n \in \mathbb{R}} (a_1 \cdot \mathbf{v}_1 + \dots + a_n \cdot \mathbf{v}_n = 0 \implies a_1 = \dots = a_n = 0)$$

Rozdział 3

Algorytm QR

W tym rozdziale zaprezentujemy elementy teorii dokonywania rozkładów QR macierzy. Rozpocniemy od sformułowania i udowodnienia twierdzenia o istnieniu takiego rozkładu. Wiele analizowanych pozycji literaturowych sygnalizowało posiadanie dowodu poniższego twierdzenia. W naszej ocenie prezentowane tam dowody bliższe są jednak jedynie ich szkicowi. Wobec powyższego prezentujemy własne opracowanie dowodu twierdzenia o istnieniu rozkładu QR, stworzonego na podstawie szkiców omówionych w literaturze oraz pracy własnej.

Twierdzenie 3.1 (O rozkładzie QR).

Niech $A \in \mathbb{R}^{m \times n}$, gdzie $m \geq n$, której kolumny są liniowo niezależne. Istnieje wtedy jedyny rozkład QR, tzn. że istnieją takie macierze Q i R , że

$$A = QR$$

i

- macierz $Q \in \mathbb{R}^{m \times n}$ jest taka, że

$$Q^T \cdot Q = D,$$

gdzie $D = \text{diag}(d_1, d_2, \dots, d_n)$, oraz $d_k > 0$ dla $k = 1, 2, \dots, n$, oraz

- macierz $R \in \mathbb{R}^{n \times n}$ jest trójkątną górną spełniającą dodatkowo warunek

$$r_{kk} = 1$$

dla wszystkich $k = 1, 2, \dots, n$.

Uwaga 3.2.

Rozkładem QR nazwiemy również sytuację, gdy

$$A = QR$$

oraz $Q^T Q = I$ i R jest macierzą trójkątną górną, niekoniecznie z 1 na przekątnej. Powodem tego jest, że prezentowana tu postać oraz postać z twierdzenia 3.1 są sobie równoważne.

Aby udowodnić twierdzenie 3.1, zaprezentujemy potrzebną teorię (twierdzenia oraz lematy) wraz z dowodami, na której to będziemy bazować przy dowodzeniu głównego twierdzenia o rozkładzie QR.

Twierdzenie 3.3 ([5, Twierdzenie 14.12]).

Wektor \mathbf{x} jest ortogonalny do podprzestrzeni W wtedy i tylko wtedy, gdy jest ortogonalny do każdego wektora jakiegokolwiek bazy tej podprzestrzeni.

Twierdzenie 3.4 (Nierówność Bessela [5, Twierdzenie 14.22]).

Jeśli $(\mathbf{b}_1, \dots, \mathbf{b}_n)$ jest układem ortonormalnym w przestrzeni euklidesowej \mathbb{R}^k , to dla każdego wektora \mathbf{x} z przestrzeni \mathbb{R}^k spełniona jest nierówność

$$\sum_{i=1}^n \alpha_i^2 \leq \|\mathbf{x}\|^2,$$

gdzie $\alpha_i = (\mathbf{x}|\mathbf{b}_i)$. Ponadto, wektor $\mathbf{x} - \sum_{i=1}^n \alpha_i \cdot \mathbf{b}_i$ jest ortogonalny do podprzestrzeni $\text{span}(\mathbf{b}_1, \dots, \mathbf{b}_n)$.

Dowód. Niech $\mathbf{x}' = \mathbf{x} - \sum_{i=1}^n \alpha_i \cdot \mathbf{b}_i$. Oczywiście $\|\mathbf{x}'\|^2 \geq 0$. Zatem

$$\begin{aligned} 0 \leq (\mathbf{x}'|\mathbf{x}') &= (\mathbf{x} - \sum_{i=1}^n \alpha_i \cdot \mathbf{b}_i | \mathbf{x} - \sum_{i=1}^n \alpha_i \cdot \mathbf{b}_i) = (\mathbf{x}|\mathbf{x}) - 2 \cdot \sum_{i=1}^n \alpha_i \cdot (\mathbf{x}|\mathbf{b}_i) + \sum_{i=1}^n \sum_{j=1}^n \alpha_i \cdot \alpha_j \cdot (\mathbf{b}_i|\mathbf{b}_j) = \\ &= \|\mathbf{x}\|^2 - 2 \cdot \sum_{i=1}^n \alpha_i^2 + \sum_{i=1}^n \alpha_i^2 = \|\mathbf{x}\|^2 - \sum_{i=1}^n \alpha_i^2. \end{aligned}$$

Przekształcając ostatnią nierówność, otrzymujemy

$$\|\mathbf{x}\|^2 \leq \sum_{i=1}^n \alpha_i^2,$$

czyli nierówność Bessela. Niech j będzie dowolną liczbą ze zbioru $1, \dots, n$. Obliczamy $(\mathbf{x}'|\mathbf{b}_j)$.

$$(\mathbf{x}'|\mathbf{b}_j) = (\mathbf{x}|\mathbf{b}_j) - \sum_{i=1}^n \alpha_i \cdot (\mathbf{b}_i|\mathbf{b}_j) = (\mathbf{x}|\mathbf{b}_j) - \alpha_j = 0.$$

Wynika stąd, że wektor \mathbf{x}' jest ortogonalny do każdego wektora \mathbf{b}_j , jest więc, na mocy poprzedniego twierdzenia ortogonalny do podprzestrzeni $\text{span}(\mathbf{b}_1, \dots, \mathbf{b}_n)$. \square

Twierdzenie 3.5 (Twierdzenie(Grama-Schmidta)[5]).

Dla każdego układu liniowo niezależnego wektorów $(\mathbf{x}_1, \dots, \mathbf{x}_n)$ w przestrzeni euklidesowej istnieje układ ortonormalny $(\mathbf{b}_1, \dots, \mathbf{b}_n)$ taki, że

$$\text{span}(\mathbf{b}_1, \dots, \mathbf{b}_k) = \text{span}(\mathbf{x}_1, \dots, \mathbf{x}_k)$$

dla każdej liczby k ze zbioru $(1, \dots, n)$.

Dowód. Oczywiście wektory \mathbf{x}_i są niezerowe, zatem mają długość dodatnią. Niech więc

$$\mathbf{b}_1 = \frac{1}{\|\mathbf{x}_1\|} \cdot \mathbf{x}_1.$$

Wektor \mathbf{b}_1 ma długość 1 oraz oczywiście $\text{span}(\mathbf{b}_1) = \text{span}(\mathbf{x}_1)$. Załóżmy teraz, że zdefiniowaliśmy już wektory $\mathbf{b}_1, \dots, \mathbf{b}_k$ tworzące układ ortonormalny taki, że

$$\text{span}(\mathbf{b}_1, \dots, \mathbf{b}_k) = \text{span}(\mathbf{x}_1, \dots, \mathbf{x}_k),$$

gdzie $k < n$. Niech teraz

$$\mathbf{b}'_{k+1} = \mathbf{x}_{k+1} - \sum_{i=1}^k (\mathbf{x}_{k+1}|\mathbf{b}_i) \cdot \mathbf{b}_i$$

oraz

$$\mathbf{b}_{k+1} = \frac{1}{\|\mathbf{b}'_{k+1}\|} \cdot \mathbf{b}'_{k+1}.$$

Na mocy twierdzenia 3.4 wnioskujemy, że wektor \mathbf{b}_{k+1} jest ortogonalny do wszystkich wektorów \mathbf{b}_i , gdzie $i \in \{1, \dots, k\}$ oraz ma długość równą 1. Ponieważ $\mathbf{b}_{k+1} \in \text{span}(\mathbf{x}_1, \dots, \mathbf{x}_{k+1})$, więc

$$\text{span}(\mathbf{b}_1, \dots, \mathbf{b}_{k+1}) \subset \text{span}(\mathbf{x}_1, \dots, \mathbf{x}_{k+1}). \quad (3.1)$$

Zauważmy, że skoro $\mathbf{x}_{k+1} \notin \text{span}(\mathbf{x}_1, \dots, \mathbf{x}_n)$, to $\mathbf{b}_{k+1} \notin \text{span}(\mathbf{b}_1, \dots, \mathbf{b}_n)$. Zatem przestrzenie $\text{span}(\mathbf{b}_1, \dots, \mathbf{b}_n)$ i $\text{span}(\mathbf{x}_1, \dots, \mathbf{x}_n)$ mają ten sam wymiar. Przestrzenie $\text{span}(\mathbf{b}_1, \dots, \mathbf{b}_{k+1})$ i $\text{span}(\mathbf{x}_1, \dots, \mathbf{x}_{k+1})$ spełniające (3.1) i mające ten sam wymiar, muszą być równe. W ten sposób określiliśmy iteracyjnie wektory $\mathbf{b}_1, \dots, \mathbf{b}_n$, spełniające żądane warunki. \square

Lemat 3.6 (O postaci macierzowej w algorytmie Grama-Schmidta).

Powyżej omówiony algorytm Grama Schmidta jest równoważny następującym transformacjom macierzy. Załóżmy, że wektory v_1, \dots, v_n są kolumnami macierzy A oraz że są liniowo niezależne. Wtedy z twierdzenia 3.5 istnieją wektory u_1, \dots, u_n , które są transformacją wektorów v_1, \dots, v_n przez algorytm Grama-Schmidta. Niech U będzie macierzą utworzoną kolumnowo z tych wektorów u_1, \dots, u_n . Wtedy

$$U = A \cdot \begin{bmatrix} 1 & -\frac{(v_2; u_1)}{(u_1; u_1)} & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -\frac{(v_3; u_1)}{(u_1; u_1)} & \dots & 0 \\ 0 & 1 & -\frac{(v_3; u_2)}{(u_2; u_2)} & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix} \cdots \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & -\frac{(v_n; u_1)}{(u_1; u_1)} \\ 0 & 1 & 0 & \dots & 0 & -\frac{(v_n; u_2)}{(u_2; u_2)} \\ 0 & 0 & 1 & \dots & 0 & -\frac{(v_n; u_3)}{(u_3; u_3)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & -\frac{(v_n; u_{n-1})}{(u_{n-1}; u_{n-1})} \\ 0 & 0 & 0 & \dots & 0 & 1 \end{bmatrix}$$

Dowód powyższego faktu jest oczywisty.

Lemat 3.7 (Macierz transformująca algorytmu Grama-Schmidta).

Niech $A \in \mathbb{R}^{m \times n}$ będzie macierzą, której kolumny są liniowo niezależne. Niech $U \in \mathbb{R}^{m \times n}$ będzie macierzą uzyskaną poprzez połączenie jako kolumn wektorów uzyskanych z algorytmu Grama-Schmidta. Wtedy istnieje trójkąta górna macierz $T \in \mathbb{R}^{n \times n}$ taka, że

$$U = A \cdot T,$$

gdzie $T = [t_{ij}]_{i=1, \dots, n}^{j=1, \dots, n}$ i $t_{kk} = 1$, dla dowolnego $k \in \{1, \dots, n\}$.

Dowód. Niech A, U takie jak w założeniach lematu. Wtedy wobec lematu 3.6 zachodzi

$$U = A \cdot \begin{bmatrix} 1 & -\frac{(v_2; u_1)}{(u_1; u_1)} & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -\frac{(v_3; u_1)}{(u_1; u_1)} & \dots & 0 \\ 0 & 1 & -\frac{(v_3; u_2)}{(u_2; u_2)} & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix} \cdots \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & -\frac{(v_n; u_1)}{(u_1; u_1)} \\ 0 & 1 & 0 & \dots & 0 & -\frac{(v_n; u_2)}{(u_2; u_2)} \\ 0 & 0 & 1 & \dots & 0 & -\frac{(v_n; u_3)}{(u_3; u_3)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & -\frac{(v_n; u_{n-1})}{(u_{n-1}; u_{n-1})} \\ 0 & 0 & 0 & \dots & 0 & 1 \end{bmatrix}.$$

Zauważmy, że mnożenie macierzowe jest łączne. Możemy je mnożyć wykonując kolejne mnożenia patrząc od prawej strony. Stosując $n - 2$ krotnie lemat 2.3 otrzymujemy macierz T , która jest nieosobliwą macierzą trójkątną górną. Ponadto zauważmy, że te wszystkie macierze mają wyłączenie 1 na przekątnej, zatem macierz T również. \square

Istotną obserwacją jaką można poczynić jest taka, że macierzy tej nie można sobie w łatwy sposób od tak wyznaczyć. Aby ją uzyskać trzeba przeprowadzić cały algorytm Grama-Schmidta i dopiero po jego przeprowadzeniu macierz tę daje się jawnie wyznaczyć.

Teraz możemy przystąpić do dowodu głównego twierdzenia.

Dowód twierdzenia 3.1. Niech $A \in \mathbb{R}^{m \times n}$ będzie macierzą jak w założeniach. Wtedy z lematu 3.7 przy zastosowaniu algorytmu Grama-Schmidta otrzymujemy równość

$$U = A \cdot T,$$

gdzie $U \in \mathbb{R}^{m \times n}$, $T \in \mathbb{R}^{n \times n}$ i U składa się z kolumn będących wektorami ortogonalnymi, a T jest macierzą trójkątną górną i ma na przekątnej same 1. Na mocy lematu 2.4 wiemy, że macierz T jest odwracalna i macierz do niej odwrotna jest trójkątną górną. Wykonując mnożenie prawostronne przez tę macierz otrzymujemy:

$$\begin{aligned} U \cdot T^{-1} &= A \cdot T \cdot T^{-1}, \\ U \cdot T^{-1} &= A. \end{aligned}$$

Postać, którą mamy powyżej jest rozkładem QR, gdzie $U = Q$ i $T^{-1} = R$. Pozostaje uzasadnić, że $U^T \cdot U = D$, gdzie $D = \text{diag}(d_1, d_2, \dots, d_n)$, oraz $d_k > 0$ dla $k = 1, 2, \dots, n$. Istotnie rozważmy element d_{ij} macierzy D . Z tego, że $U^T \cdot U = D$ mamy, że $d_{ij} = (u_i | u_j)$. Jeśli $i \neq j$, to $d_{ij} = 0$. Zatem macierz D jest przekątniowa. Jeśli $i = j$, to $d_{ii} = (u_i | u_i) = \|u_i\|^2 > 0$. □

Ortogonalizacja Grama-Schmidta może nie obliczyć macierzy ortogonalnej Q , gdy wektory które są ortogonalizowane, są prawie liniowo zależne, więc nie możemy jej użyć do stabilnego obliczenia rozkładu QR. Zamiast tego opieramy nasze algorytmy na pewnych łatwo obliczalnych ortogonalnych macierzach zwanych odbiciami Householdera lub rotacjami Givensa. Możemy je wybrać, tak aby "wprowadzić" zera do macierzy za pomocą mnożenia.

3.1 Algorytm QR metodą odbić Householdera

Metoda Householdera pozwala znaleźć rozkład QR dowolnej macierzy prostokątnej m na n ($m \geq n$).

Definicja 3.8 (Macierz Householdera).

Macierzą Householdera H , zwaną również refleksją, nazywamy macierz postaci

$$H = I - 2 \cdot \mathbf{v} \cdot \mathbf{v}^T,$$

gdzie $\|\mathbf{v}\|_2 = 1$.

Twierdzenie 3.9 (Transformacja Householdera).

Niech $\mathbf{v} \in \mathbb{R}^m$, i $\mathbf{v} \neq 0$. Wówczas transformacją Householdera nazywamy macierz postaci:

$$H = I - W \mathbf{v} \mathbf{v}^T,$$

gdzie

$$W = \frac{2}{\mathbf{v}^T \mathbf{v}}.$$

Macierz H jest macierzą symetryczną i ortogonalną. Jej interpretacja jest następująca: dowolny wektor \mathbf{x} wymiaru m jest odbiciem lustrzanym wektora $H\mathbf{x}$ względem hiperpłaszczyzny (wymiaru $m - 1$) prostopadłej do wektora \mathbf{v} .

Dowód. Łatwo sprawdzić, że tak jest ponieważ:

$$H^T = \left(I - \frac{2\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T \mathbf{v}} \right)^T = I - \left(\frac{2\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T \mathbf{v}} \right)^T = I - \frac{2}{\mathbf{v}^T \mathbf{v}} (\mathbf{v}\mathbf{v}^T)^T = I - \frac{2}{\mathbf{v}^T \mathbf{v}} (\mathbf{v}^T)^T \mathbf{v}^T = I - \frac{2}{\mathbf{v}^T \mathbf{v}} \mathbf{v}\mathbf{v}^T = H$$

oraz

$$\begin{aligned} H^2 &= \left(I - \frac{2\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}} \right)^2 = I - \frac{4\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}} + 4 \left(\frac{\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}} \right)^2 = I - \frac{4\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}} + 4 \frac{\mathbf{v}\mathbf{v}^T\mathbf{v}\mathbf{v}^T}{(\mathbf{v}^T\mathbf{v})^2} \\ &= I - \frac{4\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}} + \frac{4}{(\mathbf{v}^T\mathbf{v})^2} \mathbf{v}^T\mathbf{v}(\mathbf{v}\mathbf{v}^T) = I - \frac{4\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}} + \frac{4\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}} = I. \end{aligned}$$

Z pierwszej równości wynika symetria, z drugiej ortogonalność, ponieważ

$$H^T H = H H = I.$$

□

Obserwacja 3.10.

Macierz postaci:

$$\left[\begin{array}{c|c} 1 & 0 \\ \hline 0 & P \end{array} \right],$$

gdzie P jest transformacją Householdera, jest macierzą ortogonalną.

Przykład 3.11.

Pokażemy schemat, jak obliczyć rozkład QR macierzy A o wymiarach 5 na 4 przy użyciu transformacji Householdera. Ten przykład uwidoczni wzór dla ogólnych macierzy m na n . W poniższych macierzach P_i jest macierzą ortogonalną 5 na 5, x oznacza ogólny niezerowy wpis, a 0 oznacza zero.

1. Wyznaczamy P_1 tak, że:

$$A_1 \equiv P_1 \cdot A = \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \end{bmatrix}.$$

2. Wyznaczamy

$$P_2 = \left[\begin{array}{c|c} 1 & 0 \\ \hline 0 & P'_2 \end{array} \right]$$

tak, że

$$A_2 \equiv P_2 \cdot A_1 = \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \\ 0 & 0 & x & x \\ 0 & 0 & x & x \end{bmatrix}.$$

3. Wyznaczamy

$$P_3 = \left[\begin{array}{cc|c} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \hline 0 & 0 & P'_3 \end{array} \right]$$

tak, że

$$A_3 \equiv P_3 \cdot A_2 = \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \\ 0 & 0 & 0 & x \\ 0 & 0 & 0 & x \end{bmatrix}.$$

4. Wyznaczamy

$$P_4 = \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & P'_4 \end{array} \right]$$

tak, że

$$A_3 \equiv P_3 \cdot A_2 = \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \\ 0 & 0 & 0 & x \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

Tutaj wybraliśmy macierz Householdera P'_i aby wyzerować wyrazy w kolumnie i . Jednocześnie nie zakłóca to zer wprowadzonych już w poprzednich kolumnach. Istotnie, jeśli macierz A_k ma postać

$$\begin{bmatrix} T & A_{1k} \\ 0 & A_{2k} \end{bmatrix},$$

gdzie T oznacza macierz trójkątną górną, 0 oznacza macierz złożoną z zer, a A_{1k} i A_{2k} są macierzami o dowolnych wyrazach, to wtedy w mnożeniu dostajemy:

$$A'_{k+1} = \begin{bmatrix} I & 0 \\ 0 & P'_k \end{bmatrix} \cdot \begin{bmatrix} T & A_{1k} \\ 0 & A_{2k} \end{bmatrix} = \begin{bmatrix} T & A_{1k} \\ 0 & P'_k \cdot A_{2k} \end{bmatrix}.$$

Stąd widać, że zera pod przekątną są faktycznie zachowane.

Poniżej prezentujemy ogólny algorytm dekompozycji QR z wykorzystaniem transformacji Householdera.

Algorithm 1 Algorytm QR metodą transformacji Householdera

```

for  $i$  from 1 to  $\min(m-1, n)$  do
     $u_i = \text{House}(A(i : m, i))$ 
     $P'_i = I - 2 \cdot u_i \cdot u_i^T$ 
     $A(i : m, i : n) = P'_i \cdot A(i : m, i : n)$ 
end for

```

Implementację oraz opracowane przykłady prezentujemy w kolejnym rozdziale.

3.2 Algorytm QR metodą rotacji Givensa

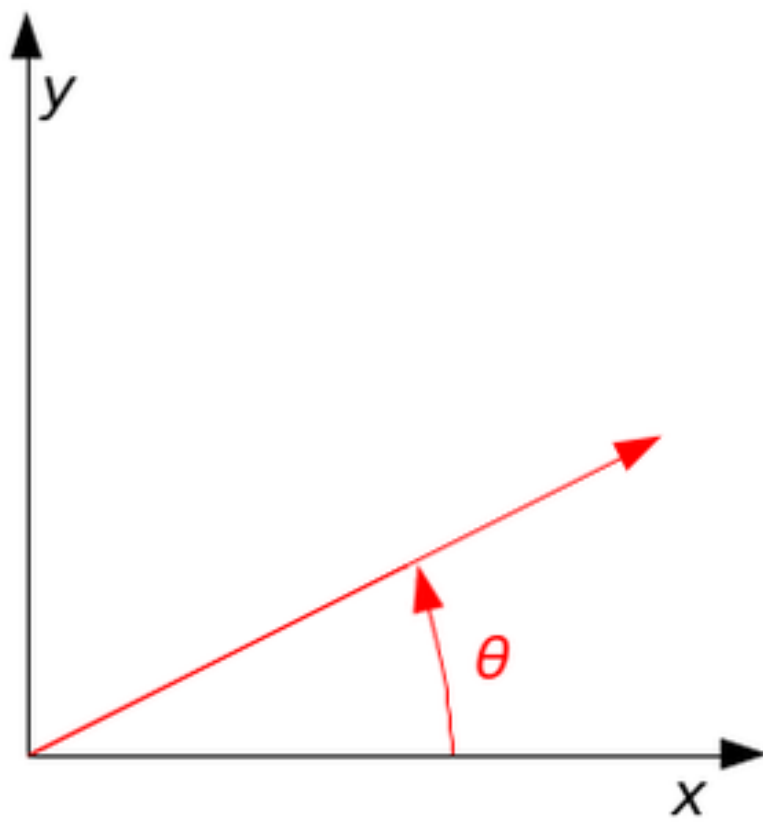
Uwaga 3.12.

Macierz postaci:

$$R(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

obraca dowolny wektor $\vec{x} \in \mathbb{R}^2$ zgodnie z ruchem wskazówek zegara o θ . Obrót ten prezentuje grafika 3.1.

Powyższa uwaga pozwala na zdefiniowanie całych grup macierzy obrotów.



Rysunek 3.1: Rotacja Givensa [2]

Definicja 3.13 (Macierz Givensa).

Niech $i, j \in 1, \dots, n$ i $\theta \in \mathbb{R}$. Macierz $R(i, j, \theta) \in \mathbb{R}^{n \times n}$ zdefiniowana następująco:

$$R(i, j, \theta) = \begin{bmatrix} 1 & 0 & \dots & 0 & \dots & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & \dots & 0 & \dots & 0 & 0 \\ 0 & 0 & \ddots & 0 & \dots & 0 & \dots & 0 & 0 \\ \vdots & \vdots & \dots & \cos \theta & \dots & -\sin \theta & \dots & 0 & 0 \\ 0 & 0 & \dots & \vdots & \ddots & \vdots & \dots & 0 & 0 \\ \vdots & \vdots & \dots & \sin \theta & \dots & \cos \theta & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 & \dots & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & \dots & \dots & \dots & \dots & \dots & 1 & 0 \\ 0 & 0 & \dots & 0 & \dots & 0 & \dots & 0 & 1 \end{bmatrix}$$

nazywamy macierzą rotacji Givensa.

Lemat 3.14 (O macierzy Givensa).

Każda macierz Givensa jest macierzą ortogonalną.

Dowód. Niech $n \in \mathbb{N}$, rozważmy macierze Givensa rozmiaru n na n . Niech $i, j \in \{1, \dots, n\}$ i $\theta \in \mathbb{R}$. Pokażemy, że macierz $R(i, j, \theta) \in \mathbb{R}^{n \times n}$ jest ortogonalna. Wystarczy pokazać, że układ kolumn (r_s) , $s = 1, \dots, n$ tej macierzy jest układem ortonormalnym. Weźmy dwie kolumny k -tą i l -tą. Rozważmy przypadki:

1. Gdy $k \notin (i, j)$, $l \notin (i, j)$, $k \neq l$, wtedy

$$\begin{aligned} (\mathbf{r}_k | \mathbf{r}_l) &= ([0, \dots, 0, 1, 0, \dots, 0] | [0, \dots, 0, 0, 1, 0, \dots, 0]) = 0, \\ (\mathbf{r}_k | \mathbf{r}_k) &= ([0, \dots, 0, 1, 0, \dots, 0] | [0, \dots, 0, 1, 0, \dots, 0]) = 1. \end{aligned}$$

2. $k \in (i, j)$, $l \in (i, j)$, $k \neq l$

$$\begin{aligned} (\mathbf{r}_i | \mathbf{r}_j) &= ([0, \dots, 0, \cos \theta, 0, \dots, 0, \sin \theta, 0, \dots, 0] | [0, \dots, 0, -(\sin \theta), 0, \dots, 0, \cos \theta, 0, \dots, 0]) = \cos \theta \cdot (-\sin \theta) + \sin \theta \cdot \cos \theta = 0, \\ (\mathbf{r}_i | \mathbf{r}_i) &= ([0, \dots, 0, \cos \theta, 0, \dots, 0, \sin \theta, 0, \dots, 0] | [0, \dots, 0, \cos \theta, 0, \dots, 0, \sin \theta, 0, \dots, 0]) = \cos^2 \theta + \sin^2 \theta = 1, \\ (\mathbf{r}_j | \mathbf{r}_j) &= ([0, \dots, 0, -(\sin \theta), 0, \dots, 0, \cos \theta, 0, \dots, 0] | [0, \dots, 0, -(\sin \theta), 0, \dots, 0, \cos \theta, 0, \dots, 0]) = -\sin^2 \theta + \cos^2 \theta = 1. \end{aligned}$$

3. $k = j$, $l \notin (i, j)$

$$(\mathbf{r}_j | \mathbf{r}_l) = (\mathbf{r}_l | \mathbf{r}_j) = ([0, \dots, 0, 1, 0, \dots, 0] | [0, \dots, 0, -(\sin \theta), 0, \dots, 0, \cos \theta, 0, \dots, 0]) = 0,$$

4. $k = i$, $l \notin (i, j)$

$$(\mathbf{r}_i | \mathbf{r}_l) = (\mathbf{r}_l | \mathbf{r}_i) = ([0, \dots, 0, 1, 0, \dots, 0] | [0, \dots, 0, \cos \theta, 0, \dots, 0, \sin \theta, 0, \dots, 0]) = 0.$$

Z uwagi na symetrię oznaczeń, również przypadek gdy $k \notin i, j$, $l = i \vee l = j$, to ich iloczyn skalarny jest równy 0. Podsumowując powyższe przypadki, z dowolności wyboru kolumn k -tej i l -tej układ kolumn $r(s)$ jest ortonormalny. Zatem macierz Givensa jest ortogonalna. \square

Lemat 3.15.

Niech $A = [a_{ij}] \in \mathbb{R}$, $i, j \in \{1, \dots, n\}$ oraz $\sqrt{a_{ii}^2 + a_{ji}^2} > 0$. Wtedy macierz Givensa $R(i, j, \theta)$, gdzie θ spełnia warunki:

$$\begin{aligned} \cos \theta &= \frac{a_{ii}}{\sqrt{a_{ii}^2 + a_{ji}^2}} \\ \sin \theta &= \frac{-(a_{ji})}{\sqrt{a_{ii}^2 + a_{ji}^2}} \end{aligned}$$

powoduje, że jeśli $[b_{ij}]_{i=1, \dots, n}^{j=1, \dots, n} = B = R(i, j, \theta) \cdot A$, to $b_{ji} = 0$.

Dowód. Niech $A \in \mathbb{R}^{n \times n}$, $i, j \in \{1, \dots, n\}$, $\sqrt{a_{ii}^2 + a_{ji}^2} > 0$ i θ spełniająca warunki. Niech $B = R(i, j, \theta) \cdot A$. Wtedy

$$b_{ji} = \sin \theta \cdot a_{ii} + \cos \theta \cdot a_{ji} = \frac{-(a_{ji})}{\sqrt{a_{ii}^2 + a_{ji}^2}} \cdot a_{ii} + \frac{a_{ii}}{\sqrt{a_{ii}^2 + a_{ji}^2}} \cdot a_{ji} = 0.$$

□

Zatem macierz $R(i, j, \theta)$ można użyć do "generowania zer w macierzy".

Algorithm 2 Algorytm QR metodą rotacji Givensa

```

for  $i$  from 1 to  $\min(m - 1, n)$  do
  for  $j$  from  $i + 1$  to  $m$  do
     $G = \text{Givens}(i, j, A[i, i], A[j, i])$ 
     $A = G \cdot A$ 
  end for
end for

```

Obserwacja 3.16.

Zauważmy jak zachowuje się macierz przy mnożeniu przez macierz Givensa.

$$\begin{bmatrix} I & 0 & 0 & 0 & 0 \\ 0 & \cos \theta & 0 & -\sin \theta & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & \sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 0 & I \end{bmatrix} \cdot \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} \\ A_{21} & A_{22} & A_{23} & A_{24} & A_{25} \\ A_{31} & A_{32} & A_{33} & A_{34} & A_{35} \\ A_{41} & A_{42} & A_{43} & A_{44} & A_{45} \\ A_{51} & A_{52} & A_{53} & A_{54} & A_{55} \end{bmatrix} =$$

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} & A_{15} \\ \cos \theta \cdot A_{21} - \sin \theta \cdot A_{41} & \cos \theta \cdot A_{22} - \sin \theta \cdot A_{42} & \cos \theta \cdot A_{23} - \sin \theta \cdot A_{43} & \cos \theta \cdot A_{24} - \sin \theta \cdot A_{44} & \cos \theta \cdot A_{25} - \sin \theta \cdot A_{45} \\ A_{31} & A_{32} & A_{33} & A_{34} & A_{35} \\ \sin \theta \cdot A_{21} + \cos \theta \cdot A_{41} & \sin \theta \cdot A_{22} + \cos \theta \cdot A_{42} & \sin \theta \cdot A_{23} + \cos \theta \cdot A_{43} & \sin \theta \cdot A_{24} + \cos \theta \cdot A_{44} & \sin \theta \cdot A_{25} + \cos \theta \cdot A_{45} \\ A_{51} & A_{52} & A_{53} & A_{54} & A_{55} \end{bmatrix}$$

Obserwacja 3.17.

Na podstawie obserwacji 3.16 możemy wywnioskować następujące:

1. Wykonanie rotacji macierzą $G(i, j, A[i, i], A[j, i])$ zachowuje zera w kolumnie i .
2. Wykonanie rotacji macierzą $G(i, j, A[i, i], A[j, i])$ zachowuje zera w kolumnach $[1 : i - 1]$ w wierszach różnych od i oraz j .

Szkic dowodu. Krótko uzasadnimy 3.17.

1. Istotnie z obserwacji 3.16 po wykonaniu obrotu Givensa względem i i j inne zera będą reprezentowane przez pola A_{12}, A_{32}, A_{52} i w efekcie nie ulegają zmianie.
2. Istotnie z obserwacji 3.16 po wykonaniu obrotu Givensa względem i i j w kolumnach $[1 : i - 1]$ w wierszach różnych od i i j są reprezentowane przez pola A_{11}, A_{31}, A_{51} .

□

Obserwacja 3.18.

Wykonanie rotacji macierzą $G(i, j, A[i, i], A[j, i])$ zachowuje zera w wierszach i, j na miejscach $A[i, 1 : i], A[j, 1 : i]$.

Szkic dowodu. Istotnie założmy, że na miejscach $A[i, 1 : i]$, $A[j, 1 : i]$ są zera. Odpowiada to polom A_{21} , A_{41} . Wtedy po rotacji mamy

$$\bar{A}_{21} = \cos \theta A_{21} - \sin \theta A_{41} = 0,$$

$$\bar{A}_{41} = \sin \theta A_{21} - \cos \theta A_{41} = 0.$$

□

Przykład 3.19.

Poniżej zilustrujemy etapy obliczania rozkładu QR macierzy 4 na 4 przy użyciu rotacji Givensa. Dla $i = 1$

$$\begin{bmatrix} x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ x & x & x & x \end{bmatrix} \xrightarrow{G(1,2)} \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ x & x & x & x \\ x & x & x & x \end{bmatrix} \xrightarrow{G(1,3)} \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ x & x & x & x \end{bmatrix} \xrightarrow{G(1,4)} \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \end{bmatrix}$$

Dla $i = 2$

$$\begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \\ 0 & x & x & x \end{bmatrix} \xrightarrow{G(2,3)} \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \\ 0 & x & x & x \end{bmatrix} \xrightarrow{G(2,4)} \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \\ 0 & 0 & x & x \end{bmatrix}$$

Dla $i = 3$

$$\begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \\ 0 & 0 & x & x \end{bmatrix} \xrightarrow{G(3,4)} \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \\ 0 & 0 & 0 & x \end{bmatrix}$$

Na mocy obserwacji 3.17 i 3.18 widzimy, że wykonanie obrotu Givensa zgodnie z zaprezentowanym wyżej przykładem zachowuje odpowiednie zera w macierzy. Z książki Demmela [4] możemy uzyskać informację, że koszt rozkładu QR za pomocą rotacji Givensa jest dwa razy większy niż koszt rozkładu za pomocą metody Householdera.

Rozdział 4

Eksperymenty numeryczne

4.1 Algorytm dekompozycji QR metodą odbić Householdera

Algorytm dekompozycji QR metodą odbić Householera posiada dwie główne składowe. Pierwszym z nich jest funkcja odnajdująca odpowiednie odbicie. Odbicie reprezentowane jest przez wektor normalny do płaszczyzny odbicia. Zadaniem jest skonstruowanie odbicia, które umożliwia "wyzerowanie" elementów położonych poniżej przekątnej w macierzy. Kod odpowiadający poszukiwaniu takiego wektora prezentujemy poniżej:

```
> house = function(x) {  
+   norm_x = sqrt(t(x) %% x)  
+   if (x[1] > 0) {  
+     x[1] = x[1] + norm_x;  
+   } else {  
+     x[1] = x[1] - norm_x;  
+   }  
+   return(x)  
+ }
```

Wtedy algorytm rozkładu QR może być zapisany za pomocą poniższego kodu:

```
> qr_householer = function(A)  
+ {  
+   R = A  
+   dimm = dim(A)  
+   n = dimm[2]  
+   m = dimm[1]  
+   Q = diag(n)  
+   for (i in 1:min(m-1,n)){  
+     u = house(R[i:m,i])  
+     den = (t(u) %% u)[1,1]  
+     P = diag(n+1-i) - 2/den * ( u%% t(u) )  
+     R[i:m,i:n] = P %% R[i:m,i:n]  
+     if (i>1)  
+       Q[i:n,1:(i-1)] = P %% Q[i:n,1:(i-1)]  
+     Q[i:n,i:n] = P %% Q[i:n,i:n]  
+   }  
+ }
```

```
+   return(list(R=R, Q=t(Q)))
+ }
```

Algorytm rozpoczynamy od zadeklarowania wymiarów rozkładanej macierzy. Następnie iteracyjnie wyznaczamy odbicia reprezentowane przez wektory normalne, które są wykorzystywane do obliczenia macierzy P , dzięki której obliczamy macierz trójkątną górną R oraz macierz Q . Warto nadmienić również, że w jednej iteracji dokonujemy wyzerowania wszystkich elementów danej kolumny pod przekątną. Cały algorytm zwraca nam macierz R oraz co ważne zwraca nam transpozycję macierzy Q .

4.2 Algorytm dekompozycji QR metodą rotacji Givensa

Algorytm dekompozycji QR metodą rotacji Givensa podobnie jak w metodzie odbić Householdera również posiada dwie główne składowe. Pierwszą z nich jest funkcja odnajdująca odpowiednią rotację, która reprezentowana jest przez macierz Givensa. Zadaniem jest skonstruowanie takiej macierzy rotującej, która umożliwia "wyzerowanie" w kolejnych kolumnach elementów położonych poniżej przekątnej w macierzy. Kod odpowiadający poszukiwaniu takiej macierzy prezentujemy poniżej:

```
> givens = function(i,j,A){
+   dimm = dim(A)
+   n = dimm[1]
+   G = diag(n)
+   p = sqrt((A[i,i])^2 + (A[j,i])^2)
+   G[i,i] = A[i,i]/ p
+   G[i,j] = (A[j,i])/p
+   G[j,i] = -(A[j,i])/p
+   G[j,j] = A[i,i]/ p
+
+   return(G)
+ }
```

Teraz tworzymy algorytm dekompozycji macierzy QR metodą rotacji Givensa:

```
> qr_givens = function(A){
+   R = A
+
+   n = dim(A)[2]
+   m = dim(A)[1]
+   Q = diag(n)
+   for (i in 1:(n-1)) {
+     for (j in (i+1):m) {
+       G = givens(i,j,R)
+       R = G %*% R
+       Q = Q %*% t(G)
+     }
+   }
+   return(list(R=R, Q=Q))
+ }
```

Algorytm rozpoczynamy od zadeklarowania wymiarów rozkładanej macierzy. Następnie iteracyjnie wyznaczamy macierz rotacji Givensa, dzięki której obliczamy macierz trójkątną górną R oraz macierz Q . Warto nadmienić również, że w jednej iteracji tego algorytmu dokonujemy wyzerowania dokładnie jednego elementu danej kolumny pod przekątną. Stąd liczba iteracji jest znacznie większa niż w algorytmie qr householder. Cały algorytm zwraca nam macierz R oraz w przeciwieństwie do algorytmu qr householder zwraca nam macierz Q .

4.3 Eksperymenty numeryczne - porównanie jakości dwóch algorytmów

Teraz czas na eksperymenty, które sprawdzą, jak bardzo dokładność naszych algorytmów różni się od gotowej funkcji R, służącej do rozkładu QR. Do wyżej wymienionych eksperymentów wykorzystamy losowo wybrane trzy macierze.

1. Wymiarów 5 na 5,
2. Wymiarów 25 na 25, oraz
3. Wymiarów 125 na 125.

4.3.1 Test dla macierzy małych rozmiarów

Na początek zajmijmy się macierzą wymiarów 5 na 5. Zostanie ona wygenerowana z losowych liczb.

```
> set.seed(2019)
> M_1<-matrix(c(replicate(5, rnorm(5))), 5, 5, byrow=TRUE)
```

Najpierw przeprowadzimy rozkład opracowanym samodzielnie algorytmem qr givens.

```
> wynik1_givens = qr_givens(M_1)
```

Następnie przeprowadzimy rozkład również opracowanym samodzielnie algorytmem qr householder.

```
> wynik1_householder = qr_householder(M_1)
```

Pozostało przeprowadzić rozkład wbudowaną funkcją w pakiet R.

```
> QR = qr(M_1)
```

Należy sprawdzić, który rozkład jest najbliższy naszej wyjściowej macierzy A . Policzymy kolejno błędy bezwzględne z:

1. Rozkładu QR metodą rotacji Givensa.
2. Rozkładu QR metodą odbić Householdera.
3. Rozkładu QR funkcją wbudowaną w język R.

- 1.

```
> M_1_givens = wynik1_givens$Q %*% wynik1_givens$R
> M_11 = M_1_givens - M_1
> blad_givens_M_1 = norm(M_11)
> blad_givens_M_1
```

```
[1] 2.331468e-15
```

2.

```
> M_1_householder = wynik1_householder$Q %*% wynik1_householder$R
> M_13 = M_1_householder - M_1
> blad_householder_M_1 = norm(M_13)
> blad_householder_M_1
```

```
[1] 3.996803e-15
```

3.

```
> M_1_qr = qr.Q(QR) %*% qr.R(QR)
> M_12 = M_1_qr - M_1
> blad_qr_M_1 = norm(M_12)
> blad_qr_M_1
```

```
[1] 1.998401e-15
```

Widzimy, że gotowy algorytm, wbudowany w R jest jednak dokładniejszy od naszych, samodzielnie napisanych. Widać również, że algorytm dekompozycji QR metodą Givensa jest lepszy od algorytmu metodą odbić Householdera. Zauważmy jednak, że nasze implementacje, nie odstają znacznie od wbudowanego w język R.

Sprawdźmy czy ta zależność znajduje zastosowania dla macierzy większych rozmiarów.

4.3.2 Test dla macierzy średniego rozmiaru

Kolej na macierz wymiarów 25 na 25.

```
> M_2<-matrix(c(replicate(25, rnorm(25))), 25, 25, byrow=TRUE)
```

Rozkład algorytmem qr givens.

```
> wynik2_givens = qr_givens(M_2)
```

Rozkład algorytmem qr householder.

```
> wynik2_householder = qr_householder(M_2)
```

Rozkład wbudowaną funkcją w Pakiet R.

```
> QR = qr(M_2)
```

Teraz wykonujemy dokładnie taką samą procedurę jak dla macierzy wymiarów 5 na 5. Liczymy kolejno błędy bezwzględne.

1.

```
> M_2_givens = wynik2_givens$Q %*% wynik2_givens$R
> M_21 = M_2_givens - M_2
> blad_givens_M_2 = norm(M_21)
> blad_givens_M_2
```

```
[1] 2.93654e-14
```

2.

```
> M_2_householder = wynik2_householder$Q %*% wynik2_householder$R
> M_23 = M_2_householder - M_2
> blad_householder_M_2 = norm(M_23)
> blad_householder_M_2
```

```
[1] 2.413347e-14
```

3.

```
> M_2_qr = qr.Q(QR) %*% qr.R(QR)
> M_22 = M_2_qr - M_2
> blad_qr_M_2 = norm(M_22)
> blad_qr_M_2
```

```
[1] 8.574738e-15
```

Tutaj obserwujemy, że błąd bezwzględny rozkładu wbudowaną funkcją w R jest wciąż najmniejszy, natomiast w tym przypadku rozkład algorytmem qr householder jest dokładniejszy od algorytmu qr givens, czyli zależność z testu dla macierzy małych rozmiarów jest odwrócona.

4.3.3 Test dla macierzy dużych rozmiarów

Sprawdzimy dokładność algorytmów dla macierzy rozmiarów 125 na 125. Zbadamy również czy przy wzroście wymiarów macierzy wzrastają również poszczególne błędy.

Wygenerujemy więc macierz:

```
> M_3<-matrix(c(replicate(125, rnorm(125))), 125, 125, byrow=TRUE)
```

Zgodnie z procedurą jaką stosowaliśmy dla macierzy mniejszych rozmiarów, przeprowadźmy kolejno:

Rozkład algorytmem qr givens:

```
> wynik3_givens = qr_givens(M_3)
```

Rozkład algorytmem qr householder:

```
> wynik3_householder = qr_householder(M_3)
```

Rozkład funkcją qr:

```
> QR = qr(M_3)
```

Dalej policzymy błędy bezwzględne.

1.

```
> M_3_givens = wynik3_givens$Q %*% wynik3_givens$R
> M_31 = M_3_givens - M_3
> blad_givens_M_3 = norm(M_31)
> blad_givens_M_3
```

```
[1] 2.709751e-13
```

2.

```
> M_3_householder = wynik3_householder$Q %*% wynik3_householder$R
> M_33 = M_3_householder - M_3
> blad_householder_M_3 = norm(M_33)
> blad_householder_M_3
```

```
[1] 4.039776e-13
```

3.

```
> M_3_qr = qr.Q(QR) %*% qr.R(QR)
> M_32 = M_3_qr - M_3
> blad_qr_M_3 = norm(M_32)
> blad_qr_M_3
```

```
[1] 8.038709e-14
```

4.3.4 Podsumowanie

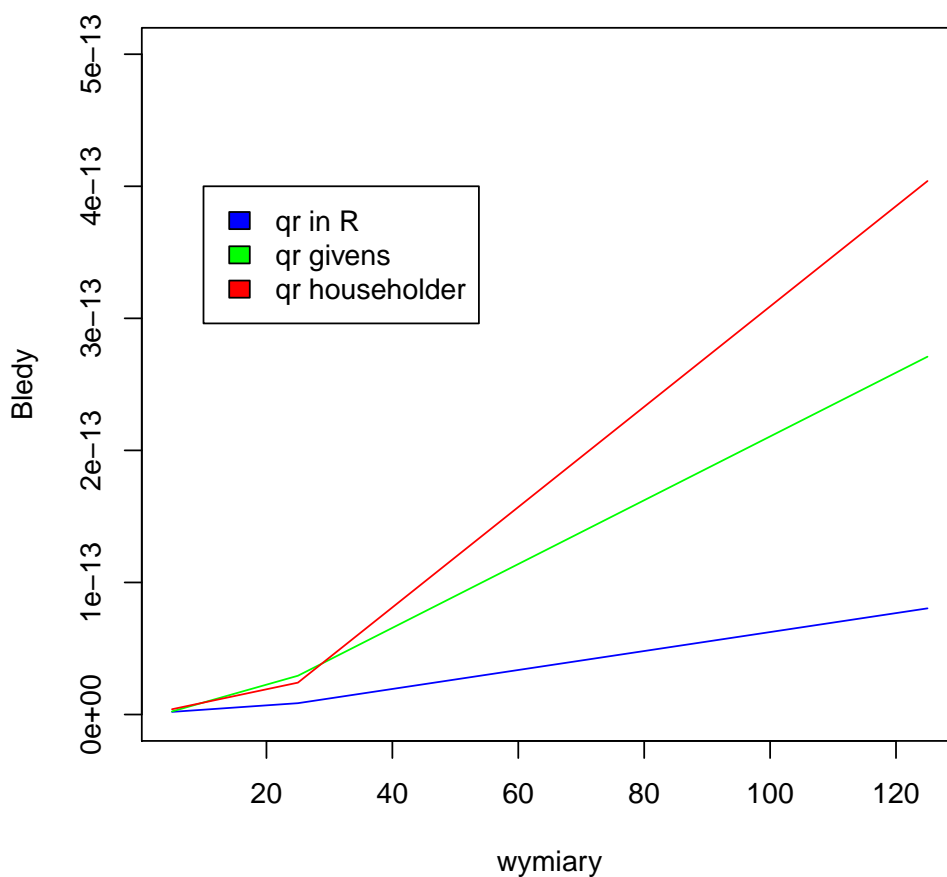
W tym momencie możemy wysnuć wniosek, że rozmiar macierzy nie wpływa istotnie na jakość naszych samodzielnie opisanych funkcji w stosunku do gotowej funkcji w R. Są one mniej dokładne niż wbudowany, gotowy algorytm dekompozycji QR w R. Ponadto algorytm metodą rotacji Givensa nie zawsze jest jakościowo lepszy od algorytmu metodą odbić Householdera. Zależy to od wymiarów oraz od losowości rozkładanej macierzy.

Zbadamy teraz jak zachowują się błędy w poszczególnych algorytmach przy zwiększaniu wymiaru. Interesuje nas czy te błędy rosną wraz ze wzrostem wymiaru, czy też wręcz przeciwnie.

W tym celu stworzymy wykres prezentujący jak zachowują się błędy bezwzględne trzech przedstawionych przez nas algorytmów dla macierzy wymiarów omówionych przez nas.

```
> wymiary<-c(5,25,125)
> bledy1<-c(blad_qr_M_1,blad_qr_M_2,blad_qr_M_3)
> bledy2<-c(blad_givens_M_1,blad_givens_M_2,blad_givens_M_3)
> bledy3<-c(blad_householder_M_1,blad_householder_M_2,blad_householder_M_3)

> plot(wymiary, bledy1, type = 'l', col = 'blue', ylim=c(0,5e-13), ylab = "Bledy")
> lines(wymiary, bledy2, col='green')
> lines(wymiary, bledy3, col='red')
> legend(10,4e-13, c("qr in R", "qr givens", "qr householder"),
+       fill=c("blue", "green", "red"))
```



Wyniki nie są zaskakujące. Wraz ze wzrostem wymiaru rosną również błędy bezwzględne wszystkich przedstawionych przez nas algorytmów. Z wykresu można również zaobserwować, iż ten wzrost jest więcej niż liniowy. Warto również nadmienić, że błąd algorytmu QR metodą odbić Householdera rośnie najszybciej, zaś tempo wzrostu błędu wbudowanej funkcji jest najniższe.

Rozdział 5

Podsumowanie

W pracy tej zajmowaliśmy się tematyką dotyczącą algorytmu QR . W rozdziale 3 udało się zaprezentować uzupełniony dowód istnienia rozkładu QR dowolnej macierzy. Zaprezentowany dowód opiera się w głównej mierze o twierdzenie o ortogonalizacji Grama-Schmidta, przy wykorzystaniu kilku podstawowych faktów, dotyczących macierzy trójkątnych górnych i ortogonalnych. Dalej w sekcjach 3.1 oraz 3.2 prezentujemy i omawiamy stabilnie numerycznie algorytmy znajdowania dekompozycji QR . W sekcji 3.1 algorytm QR metodą odbić Householdera, dla którego pokazujemy kilka faktów dotyczących własności macierzy odbić oraz uzasadniamy dlaczego w metodzie tej zachowane są zera w macierzy w czasie procesu iterowania. W sekcji 3.2 prezentujemy algorytm QR metodą rotacji Givensa, wykorzystujący macierze rotacji. Również w tym wypadku dowodzimy prostych faktów dla tychże macierzy, a za pomocą obserwacji 3.16, 3.17, 3.18 przedstawiamy czytelnikowi szkic argumentacji o zachowaniu zer w macierzy w procesie iterowania. W rozdziale 4 pracy zaprezentowaliśmy własne implementacje algorytmów, omówionych w rozdziale 3. W sekcji 4.1 znajduje się implementacja algorytmu QR metodą odbić z wykorzystaniem składni języka R. Czytelnik znajdzie natomiast w sekcji 4.2 implementację drugiego z algorytmów, algorytmu QR metodą rotacji. Wreszcie w sekcji 4.3 przeprowadzamy kilka prostych eksperymentów z wykorzystaniem naszych implementacji. Przeprowadzamy testy dla macierzy o rozmiarach:

1. 5 na 5
2. 25 na 25
3. 125 na 125

porównując je ze sobą oraz z wbudowaną implementacją w języku R.

W efekcie przeprowadzonych eksperymentów stwierdzamy, że nasze własne implementacje niestety ustępują wbudowanej funkcji. Godnym odnotowaniem jest jednak fakt, iż uzyskane przez nie wyniki są jakościowo bliskie narzędziu wbudowanemu. Stosowanie naszych algorytmów nie powoduje bowiem wprowadzenia niedokładności większych niż 10^{-13} dla macierzy o rozmiarach nawet 125 na 125.

Bibliografia

- [1] Dokumentacja online języka r. <https://www.rdocumentation.org>.
- [2] Ilustracja rotacji givensa. https://pl.wikipedia.org/wiki/Macierz_obrotu#/media/Plik:Counterclockwise_rotation.png.
- [3] Åke Björck. *Numerical Methods for Least Squares Problems*. Siam, 1996.
- [4] James W. Demmel. *Applied Numerical Linear Algebra*. Siam, 1997.
- [5] Jacek Jędrzejewski and Tadeusz Poreda. *Algebra liniowa z elementami geometrii analitycznej*. Politechnika Łódzka, 2011.
- [6] James H. Wilkinson. *The algebraic eigenvalue problem*. Clarendon Press, Oxford University Press, 1988.