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

Rozdział 1

Wstęp

Tematem niniejszej pracy jest algorytm QR dekompozycji macierzy, który to został wynaleziony w 1961 roku przez Francisa i Kublanowską. 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 [2, 3, 4, 5]. 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 wszystkim macierzy o m wierszach i n kolumnach
•*span* - przestrzeń rozpięta na wektorach
•$(\cdot|\cdot)$ - iloczyn skalarny
•$||\cdot||$ - norma
•A, B - macierze
•x - szukana
•I - macierz identycznościowa

2.2 Elementy algebry liniowej

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

Definicja 2.1 (Macierz [4])**.**
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

$$A : \{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\} \to \mathbb{R}.$$

Wartością funkcji dla argumentu (i, j), gdzie $i \in \{1, 2, \ldots, m\}$, $j \in \{1, 2, \ldots, n\}$ 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. Wymieńmy kilka typów macierzy, szczególnie interesujących z perspektywy omawianego przez nas tematu.

Definicja 2.2 (Rodzaje macierzy[4])**.**
• *Macierzą 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 zeru, to taką macierz nazywamy macierzą diagonalną, oznaczamy ją jako* $diag(a_{11}, a_{22}, \ldots, a_{nn})$.
• *Macierz jednostkową stopnia n nazywamy taką macierz diagonalną, w której wszystkie elementy na głównej przekątnej są równe 1.*
• *Macierzą symetryczną nazywamy macierz kwadratową* $A = [a_{ij}]$*, której wyrazy spełniają warunek*

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

• *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.*

Następujące lematy zostały opracowane na potrzeby dowodu w rozdziale 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 $i, j \in \{1, \ldots, n\}$ parę indeksów poniżej przekątnej, tj. takich dla których $i > j$. Wtedy

$$c_{ij} = a_{i1}b_{1j} + \ldots + a_{in}b_{nj}. \tag{2.1}$$

Zauważmy, że elementy $a_{i1}, \ldots, a_{i(i-1)}$ są równe 0 gdyż pochodzą z macierzy trójkątnej górnej. Ponadto elementy $b_{j(j+1)}, \ldots, b_{jn}$ 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 $i \in \{1, \ldots, n\}$. Wtedy

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

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* $A^{-1} = [\overline{a}_{ij}]_{i=1,\ldots,n}^{j=1,\ldots,n}$ *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 [4] i twierdzenia 9.22 z [4] i nieosobliwości macierzy A wiemy, że macierz A^{-1} istnieje i jest nieosobliwa. Pozostaje pokazać, że jest macierzą trójkątną. Pokazać ten fakt można na kilka różnych sposobów, zaprezentujemy tutaj najprostszy pod względem wiedzy teoretycznej dowód.

Wykorzystamy drugi krok z algorytmu elimininacji 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

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

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.

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

Przypuśćmy nie wprost, że macierz A^{-1} nie jest trójkątną górną. Istnieje zatem takie $i, j \in \{1, \dots, n\}$, $i > j$ i $\overline{a}_{ij} \neq 0$. 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

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

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 $\overline{a}_{ii} \neq 1$. 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

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

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

□

Definicja 2.5.
Niech $A = [a_{ij}] \in \mathbb{R}^{m \times n}$. Wtedy macierz $D \in \mathbb{R}^{n \times m}$ spełniająca warunek taki, że

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

dla dowolnych $i \in \{1, \dots, n\}$ oraz $j \in \{1, \dots, m\}$ nazywamy macierzą transponowaną do A. Z reguły tę macierz D oznaczamy symbolem A^T.

Definicja 2.6 (Iloczyn macierzy[4])**.**
Niech $A \in \mathbb{R}^{m \times n}$ i $B \in \mathbb{R}^{m \times n}$. Jeśli

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

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 2.7 (Macierz odwrotna[4])**.**
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 2.8.
Macierzą ortogonalną nazywamy macierz kwadratową $A \in \mathbb{R}^{n\times n}$ o elementach będących liczbami rzeczywistymi spełniająca równość:

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

Lemat 2.9 (O iloczynie macierzy ortogonalnych)**.**
Niech $A, B \in \mathbb{R}^{n\times n}$ będą dwiema ortogonalnymi macierzami. Wówczas

$$C = AB$$

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

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

Zatem C jest ortogonalna. □

Definicja 2.10.
Wektorem kolumnowym nazywamy macierz z przestrzeni $\mathbb{R}^{n\times 1}$.

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

Definicja 2.11.
Niech $\mathbf{v_1}, ..., \mathbf{v_n}$ będą różnymi elementami przestrzeni liniowej $\mathbb{R}^k$. Zbiór wektorów $\mathbf{v_1}, ..., \mathbf{v_n}$ nazywamy liniowo niezależnymi jeżeli

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

Rozdział 3

Algorytm QR

W tym rozdziale zaprezentujemy elementy teorii dokonywania rozkładów QR macierzy. Rozpoczniemy 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 $m \geqslant n$, 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 $D = diag(d_1, d_2, ..., d_n)$, oraz $d_k > 0$ dla $k = 1, 2, \ldots, n$, oraz
- *macierz $R \in \mathbb{R}^{n\times n}$ jest trójkątną górną spełniającą dodatkowo warunek*
$$r_{kk} = 1$$
dla wszystkich $k = 1, 2, \ldots, n$.

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

Twierdzenie 3.2 ([4, Twierdzenie 14.12])**.**
Wektor $\mathbf{x}$ jest ortogonalny do podprzestrzeni W wtedy i tylko wtedy, gdy jest ortogonalny do każdego wektora jakiejkolwiek bazy tej podprzestrzeni.

Twierdzenie 3.3 (Nierówność Bessela [4, Twierdzenie 14.22])**.**
Jeśli $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ jest układem ortonormalnym w przestrzeni euklidesowej V, to dla każdego wektora $\mathbf{x}$ z przestrzeni V spełniona jest nierówność

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

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 $span(\mathbf{b}_1, \ldots, \mathbf{b}_n)$.

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

$$0 \leqslant (\mathbf{x}^{'}|\mathbf{x}^{'}) = (\mathbf{x} - \sum_{i=1}^{n} \alpha_i \cdot \mathbf{b}_i | \mathbf{x} - \sum_{i=1}^{n} \alpha_i \cdot \mathbf{b}_i) = (\mathbf{x}|\mathbf{x}) - 2 \cdot \sum_{i=1}^{n} \alpha_i \cdot (\mathbf{x}|\mathbf{b}_i) + \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \cdot \alpha_j \cdot (\mathbf{b}_i|\mathbf{b}_j) =$$

$$\|\mathbf{x}\|^2 - 2 \cdot \sum_{i=1}^{n} \alpha_i^2 + \sum_{i=1}^{n} \alpha_i^2 = \|\mathbf{x}\|^2 - \sum_{i=1}^{n} \alpha_i^2.$$

Przekształcając ostatnią nierówność, otrzymujemy

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

czyli nierówność Bessela. Niech j będzie dowolną liczbą ze zbioru $1, \ldots, n$. 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 $span(\mathbf{b}_1, \ldots, \mathbf{b}_n)$. □

Twierdzenie 3.4 (Twierdzenie(Grama-Schmidta)[4])**.**
Dla każdego układu liniowo niezależnego wektorów $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ *w przestrzeni euklidesowej istnieje układ ortonormalny* $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ *taki, że*

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

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

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

$$\mathbf{b}_1 = \frac{1}{\|(x_1)\|} \cdot \mathbf{x}_1.$$

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

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

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 3.3 poprzedniego wnioskujemy, że wektor $\mathbf{b}_{k+1}$ jest ortogonalny do wszystkich wektorów $\mathbf{b}_i$, gdzie $i \in \{1, \ldots, k\}$ oraz ma długość równą 1. Ponieważ $\mathbf{b}_{k+1} \in span(\mathbf{x}_1, \ldots, \mathbf{x}_{k+1})$, więc

$$span(\mathbf{b}_1, \ldots, \mathbf{b}_{k+1}) \subset span(\mathbf{x}_1, \ldots, \mathbf{x}_{k+1}). \tag{3.1}$$

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

Lemat 3.5 (O postaci macierzowej w algorytmie Grama-Schmidta)**.**
Powyżej omówiony algorytm Grama Schmidta jest równoważny następującym transformacjom macierzy. Załóżmy, że wektory $\mathbf{v_1}, \ldots, \mathbf{v_n}$ *są kolumnami macierzy* A *oraz że są liniowo niezależne. Wtedy z twierdzenia 3.4 istnieją wektory* $\mathbf{u_1}, \ldots, \mathbf{u_n}$ *, które są transformacją wektorów* $\mathbf{v_1}, \ldots, \mathbf{v_n}$ *przez algorytm Grama-Schmidta. Niech* U *będzie macierzą utworzoną kolumnowo z tych wektorów* $\mathbf{u_1}, \ldots, \mathbf{u_n}$*. Wtedy*

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

Dowód powyższego faktu jest oczywisty.

Lemat 3.6 (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 $T = [t_{ij}]_{i=1,\ldots,n}^{j=1,\ldots,n}$ *i* $t_{kk} = 1$*, dla dowolnego* $k \in \{1, \ldots, n\}$*.*

Dowód. Niech A, U takie jak w założeniach lematu. Wtedy wobec lematu 3.5 zachodzi

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

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. □

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 3.6 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 wektórów ortogonalnych, 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:

$$U \cdot T^{-1} = A \cdot T \cdot T^{-1},$$
$$U \cdot T^{-1} = A.$$

Postać, którą mamy powyżej jest rozkładem QR, gdzie $U = Q$ i $T^{-1} = R$. Pozostaje uzasadnić, że $U^T \cdot U = D$, gdzie $D = diag(d_1, d_2, ..., d_n)$, oraz $d_k > 0$ dla $k = 1, 2, \ldots, n$. 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 Gram-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 ($m \geqslant n$).

Definicja 3.7 (Macierz Householdera)**.**
Macierzą Householdera H zwaną również refleksją nazywamy macierz postaci $H = I - 2 \cdot \mathbf{v} \cdot \mathbf{v}^T$, *gdzie* $\|v\|_2 = 1$.

Twierdzenie 3.8 (Transformacja Householdera)**.**
Niech $\mathbf{v} \in R^m$, *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 x wymiaru m jest odbiciem lustrzanym wektora $\mathbf{Hx}$ *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

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

Z pierwszej równości wynika symetria, z drugiej ortogonalność, ponieważ

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

□

Przykład 3.9.
Pokażemy, 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 o 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 subdiagonale wyrazy w kolumnie i. Jednocześnie nie zakłóca to zer wprowadzonych już w poprzednich kolumnach. Istotnie macierz A_k ma postać

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

Wtedy w mnożeniu dostajemy:

$$P_k' = \begin{bmatrix} I & 0 \\ 0 & P_k' \end{bmatrix} \cdot \begin{bmatrix} T & A_{1k} \\ 0 & A_{2k} \end{bmatrix} = \begin{bmatrix} T & A_{1k} \\ 0 & P_k' \cdot A_{2k} \end{bmatrix}.$$

Stąd widać, że zera pod przekątną są faktycznie zachowane. Nazwijmy ostatnią trójkątną górną macierz 5 *na* 4 $\tilde{R} = A_4$. *Następnie*

$$A = P_1^T \cdot P_2^T \cdot P_3^T \cdot P_4^T \cdot R = QR,$$

gdzie Q *to pierwsze cztery kolumny*

$$P_1^T \cdot P_2^T \cdot P_3^T \cdot P_4^T = P_1 \cdot P_2 \cdot P_3 \cdot P_4$$

(ponieważ wszystkie P_i *są symetryczne), a* R *to pierwsze cztery rzędy* $\tilde{R}$.

Oto 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**
  $u_i = House(A(i:m, i))$
  $P_i' = I - 2 \cdot u_i \cdot u_i^T$
  $A(i:m, i:n) = P_i' \cdot A(i:m, i:n)$
end for

3.2 Algorytm QR metodą rotacji Givensa

Uwaga 3.10.
Rotacja Givensa postaci:

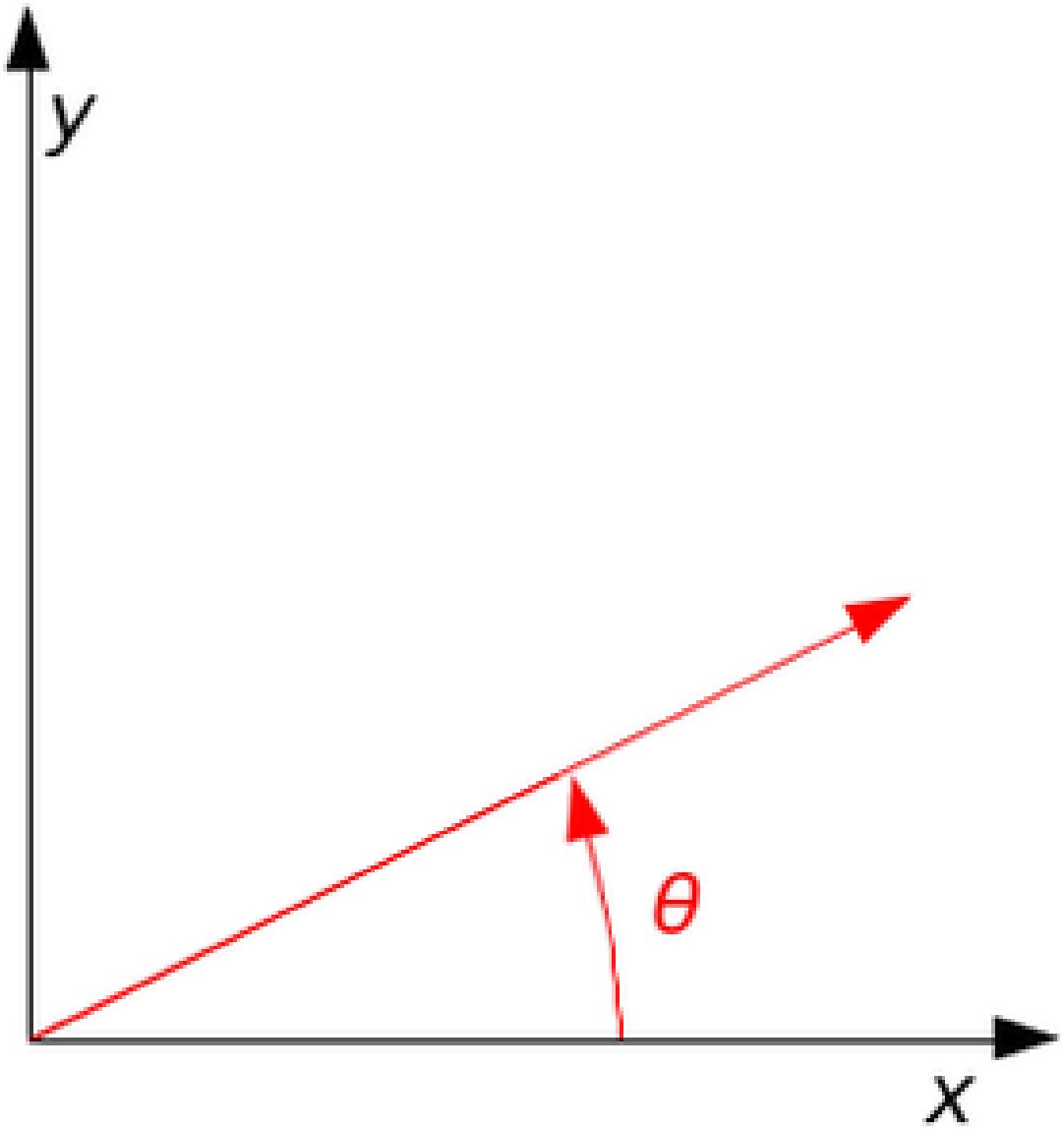
$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

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

Definicja 3.11 (Macierz Givensa)**.**
Niech $i, j \in 1, \ldots, n$ *i* $\theta \in \mathbb{R}$. *Macierz* $R(i, j, \theta) \in \mathbb{R}^{n \times n}$ *zdefiniowana następująco:*

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

nazywamy macierzą rotacji Givensa.



Rysunek 3.1: Rotacja Givensa

Lemat 3.12 (O macierzy Givensa)**.**
Każda macierz Givensa jest macierzą ortogonalną.

Dowód. Niech $n \in \mathbb{N}$, rozważmy macierze Givensa rozmiaru n na n. Niech $i, j \in 1, \ldots, n$ 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), $s = 1, \ldots, n$ 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

$$(\mathbf{r}_k|\mathbf{r}_l) = ([0, \ldots, 0, 1, 0, \ldots, 0]|[0, \ldots, 0, 0, 1, 0, \ldots, 0]) = 0$$

$$(\mathbf{r}_k|\mathbf{r}_k) = ([0, \ldots, 0, 1, 0, \ldots, 0]|[0, \ldots, 0, 1, 0, \ldots, 0]) = 1$$

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

$$(\mathbf{r}_i|\mathbf{r}_j) = ([0, \ldots, 0, \cos\theta, 0, \ldots, 0, \sin\theta, 0, \ldots, 0]|[0, \ldots, 0, -(\sin\theta), 0, \ldots, 0, \cos\theta, 0, \ldots, 0]) = \cos\theta \cdot (-\sin\theta) + \sin\theta \cdot \cos\theta = 0$$

$$(\mathbf{r}_i|\mathbf{r}_i) = ([0, \ldots, 0, \cos\theta, 0, \ldots, 0, \sin\theta, 0, \ldots, 0]|[0, \ldots, 0, \cos\theta, 0, \ldots, 0, \sin\theta, 0, \ldots, 0]) = \cos^2\theta + \sin^2\theta = 1$$

$$(\mathbf{r}_j|\mathbf{r}_j) = ([0, \ldots, 0, -(\sin\theta), 0, \ldots, 0, \cos\theta, 0, \ldots, 0]|[0, \ldots, 0, -(\sin\theta), 0, \ldots, 0, \cos\theta, 0, \ldots, 0]) = -\sin^2\theta + \cos^2\theta = 1$$

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

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

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

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

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. □

Lemat 3.13.
Niech $A \in \mathbb{R}^{n \times n}$. Wtedy macierz Givensa $R(i, j, \theta)$, gdzie θ spełnia warunki:

$$\cos\theta = \frac{a_{ii}}{\sqrt{a_{ii}^2 + a_{ji}^2}}$$

$$\sin\theta = \frac{-(a_{ji})}{\sqrt{a_{ii}^2 + a_{ji}^2}}$$

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

Dowód. Niech $A \in \mathbb{R}^{n \times n}$, $ij \in 1, \ldots, n$ 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 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 = Givens(i, j, A[i, i], A[j, i])
        A = G · A
    end for
end for
```

Obserwacja 3.14.
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 3.15.
Wykonanie rotacji macierzą $G(i, j, A[i, i], A[j, i])$ zachowuje zera w kolumnie j.

Szkic dowodu. Istotnie z obserwacji 3.14 po wykonaniu obrotu Givensa względem i i j inne zera będą reprezentowane przez pola $A_{12}, A_{32}, A_{52}, \ldots$ i nie ulegają zmianie. □

Obserwacja 3.16.
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 załóżmy, że $G(i, j, A[i, i], A[j, i])$ są wypełnione zerami. Odpowiada to polom A_{21}, A_{41}. Wtedy po rotacji mamy

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

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

□

Przykład 3.17.
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 3.15 i 3.16 widzimy, że wykonanie obrotu Givensa zgodnie z zaprezentowanym wyżej przykładem zachowuje odpowiednie zera w macierzy. Z książki "Applied Numerical Linear Algebra" autorstwa Jamesa W. Demmela 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 łatwo 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)))
+ }
```

Jest to gotowy algorytm, który zwraca nam macierz Q i macierz R.

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 łatwo 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))
+ }
```

Podobnie jak algorytm QR metodą odbić Householdera ten algorytm również zwraca nam dwie macierze Q i R

4.3 Eksperymenty numeryczne - porównanie jakości dwóch algorytmów

Teraz czas na eksperymenty, które pokażą, jak bardzo różni się błąd naszych algorytmów od tego z gotowej funkcji R Studio, 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.

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

Najpierw przeprowadzimy rozkład wbudowaną funkcją w pakiet R.

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

Następnie przeprowadzimy rozkład opracowanym samodzielnie algorytmem qr givens.

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

Pozostało przeprowadzić rozkład również opracowanym samodzielnie algorytmem qr householder.

```
> wynik1_householder = qr_householer(M_1)
```

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

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

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] 2.164935e-15
```

2.

```
> 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] 1.720846e-15
```

3.

```
> 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] 1.387779e-15
```

Sprawdźmy, który z błędów jest najmniejszy, a który największy. Będziemy wtedy w stanie ocenić, który z rozkładów jest najdokładniejszy.

```
> min(blad_givens_M_1, blad_householder_M_1, blad_qr_M_1)

[1] 1.387779e-15

> max(blad_givens_M_1, blad_householder_M_1, blad_qr_M_1)

[1] 2.164935e-15
```

Widzimy, że gotowy algorytm, wbudowany w RStudio jest jednak dokładniejszy od naszych, samodzielnie napisanych. Widać również, że algorytm dekompozycji QR metodą Givensa jest lepszy od algorytmu metodą odbić Householdera.

Sprawdzimy 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 wbudowaną funkcją w Pakiet R.

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

Rozkład algorytmem qr givens.

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

Rozkład algorytmem qr householder.

```
> wynik2_householder = qr_householer(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] 2.462613e-14
```

2.

```
> 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] 9.249546e-15
```

3.

```
> 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] 1.898481e-14
```

Tak jak w poprzednim przykładzie porównajmy uzyskane błędy.

```
> min(blad_givens_M_2,blad_householder_M_2, blad_qr_M_2)

[1] 9.249546e-15

> max(blad_givens_M_2,blad_householder_M_2, blad_qr_M_2)

[1] 2.462613e-14
```

Widzimy, że zwiększenie wymiarów macierzy nie wpływa na jakości prezentowanych algorytmów.

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.

Wygenerujmy 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 funkcją qr:

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

Rozkład algorytmem qr givens:

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

Rozkład algorytmem qr householder:

```
> wynik3_householder = qr_householer(M_3)
```

Dalej policzymy błędy bezwględne.
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] 2.273971e-13
```

2.

```
> 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] 8.031358e-14
```

3.

```
> 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] 4.263603e-13
```

Zgodnie z procedurą badamy, który z błędów jest najmniejszy, a który największy.

```
> min(blad_givens_M_3,blad_householder_M_3,blad_qr_M_3)
```

```
[1] 8.031358e-14
```

```
> max(blad_givens_M_3,blad_householder_M_3,blad_qr_M_3)
```

```
[1] 4.263603e-13
```

W tym momencie możemy wysnuć wniosek, że rozmiar macierzy nie ma znaczenia jeżeli chodzi o jakość naszych samodzielnie opisanych funkcji. Są one mniej dokładne niż wbudowany, gotowy algorytm dekompozycji QR w R Studio. Ponadto algorytm metodą rotacji Givensa jest jakościowo lepszy od algorytmu metodą odbić Householdera. 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.

Porównajmy kolejno błędy bezwzględne z:

1. rozkładu qr givens
2. rozkładu qr householder

3. rozkładu funkcją wbudowaną w język R.

1.

```
> blad_givens_M_1
[1] 2.164935e-15
> blad_givens_M_2
[1] 2.462613e-14
> blad_givens_M_3
[1] 2.273971e-13
> min(blad_givens_M_1,blad_givens_M_2)
[1] 2.164935e-15
> min(blad_givens_M_2,blad_givens_M_3)
[1] 2.462613e-14
```

2.

```
> blad_householder_M_1
[1] 1.387779e-15
> blad_householder_M_2
[1] 1.898481e-14
> blad_householder_M_3
[1] 4.263603e-13
> min(blad_householder_M_1,blad_householder_M_2)
[1] 1.387779e-15
> min(blad_householder_M_2,blad_householder_M_3)
[1] 1.898481e-14
```

3.

```
> blad_qr_M_1
[1] 1.720846e-15
> blad_qr_M_2
[1] 9.249546e-15
> blad_qr_M_3
[1] 8.031358e-14
> min(blad_qr_M_1,blad_qr_M_2)
[1] 1.720846e-15
> min(blad_qr_M_2,blad_qr_M_3)
[1] 9.249546e-15
```

Wyniki nie są zaskakujące. Wraz ze wzrostem wymiaru rosną również błędy bezwględne wszystkich przedstawionych przez nas algorytmów.

Rozdział 5

Podsumowanie

Bibliografia

[1] Dokumentacja online języka r. `https://www.rdocumentation.org`.

[2] Åke Björck. *Numerical Methods for Least Squares Problems*. Siam, 1996.

[3] James W. Demmel. *Applied Numerical Linear Algebra*. Siam, 1997.

[4] Jacek Jędrzejewski and Tadeusz Poreda. *Algebra liniowa z elementami geometrii analitycznej*. Politechnika Łodzka, 2011.

[5] James H. Wilkonson. *The algebraic eigenvalue problem*. Clarendon Press, Oxford University Press, 1988.