Lecture Note 9: Orthogonal Reduction

Xianyi Zeng
Department of Mathematical Sciences, UTEP

1 The Row Echelon Form

Our target is to solve the normal equation:

\[ A^t A x = A^t b, \]

where \( A \in \mathbb{R}^{m \times n} \) is arbitrary; we have shown previously that this is equivalent to the least squares problem:

\[ \min_{x \in \mathbb{R}^n} \| Ax - b \|. \]

A first observation we can make is that (1.1) seems familiar! As \( A^t A \in \mathbb{R}^{n \times n} \) is symmetric semi-positive definite, we can try to compute the Cholesky decomposition such that \( A^t A = L^t L \) for some lower-triangular matrix \( L \in \mathbb{R}^{n \times n} \). One problem with this approach is that we’re not fully exploring our information, particularly in Cholesky decomposition we treat \( A^t A \) as a single entity in ignorance of the information about \( A \) itself.

Particularly, the structure \( A^t A \) motivates us to study a factorization \( A = QE \), where \( Q \in \mathbb{R}^{m \times m} \) is orthogonal and \( E \in \mathbb{R}^{m \times n} \) is to be determined. Then we may transform the normal equation to:

\[ E^t E x = E^t Q^t b, \]

where the identity \( Q^t Q = I_m \) (the identity matrix in \( \mathbb{R}^{m \times m} \)) is used. This normal equation is equivalent to the least squares problem with \( E \):

\[ \min_{x \in \mathbb{R}^n} \| Ex - Q^t b \|. \]

Because orthogonal transformation preserves the \( L^2 \)-norm, (1.2) and (1.4) are equivalent to each other. Indeed, for any \( x \in \mathbb{R}^n \):

\[
\| Ax - b \|^2 = (b - Ax)^t (b - Ax) = (b - QEx)^t (b - QEx) = [Q(Q^t b - Ex)]^t [Q(Q^t b - Ex)]
= (Q^t b - Ex)^t Q Q (Q^t b - Ex) = (Q^t b - Ex)^t (Q^t b - Ex) = \| Ex - Q^t b \|^2.
\]

Hence the target is to find an \( E \) such that (1.3) is easier to solve. Motivated by the Cholesky decomposition, we’d like to find an \( E \) with a structure similar to the upper-triangular matrices.

To this end, we say that \( E \in \mathbb{R}^{m \times n} \) is of the row echelon form defined below.

**Definition 1.** Let \( E = [e_{ij}] \in \mathbb{R}^{m \times n} \) be arbitrary, we define for each row number \( 1 \leq i \leq m \) a positive number \( n_i \) such that \( e_{im} \neq 0 \) and \( e_{ij} = 0 \) for all \( j < n_i \). If the entire \( i \)-th row is zero, we set \( n_i = n + 1 \). Then the matrix \( E \) is said to have the row echelon form if and only if the sequence \( \{n_1, n_2, \ldots, n_m\} \) is strictly increasing until it reaches and stays at the value \( n + 1 \).
Graphically, such a matrix looks like:

\[
E = \begin{bmatrix}
  * & * & * & \cdots & * & * & * \\
  0 & * & * & \cdots & * & * & * \\
  0 & 0 & 0 & \cdots & * & * & * \\
  \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\
  0 & 0 & 0 & 0 & \cdots & 0 & * \\
  0 & 0 & 0 & 0 & 0 & 0 & *
\end{bmatrix}.
\]  

We will see that for a matrix of row echelon form, the least squares problem (1.4) is easy to solve. Let \( d = Q^T b \), then the residual vector is given by:

\[
Ex - d = \begin{bmatrix}
  e_{1n_1}x_{n_1} + e_{1,n_1+1}x_{n_1+1} + \cdots + e_{1n}x_n - d_1 \\
  e_{2n_2}x_{n_2} + e_{2,n_2+1}x_{n_2+1} + \cdots + e_{2n}x_n - d_2 \\
  \vdots \\
  e_{ln_l}x_{n_l} + e_{l,n_l+1}x_{n_l+1} + \cdots + e_{ln}x_n - d_l \\
  -d_{l+1} \\
  \vdots \\
  -d_m
\end{bmatrix},
\]

where \( l \) is the last non-zero row of \( E \). Note that except for the first term, all other components of the residual are independent of \( x_{n_1} \); hence we must have:

\[
x_{n_1} = \frac{1}{e_{1n_1}} \left( d_1 - \sum_{j=n_1+1}^{n} e_{1j}x_j \right). \tag{1.6}
\]

Similarly, if \( l \geq 2 \) we have \( e_{2n_2} \neq 0 \) and we deduce:

\[
x_{n_2} = \frac{1}{e_{2n_2}} \left( d_2 - \sum_{j=n_2+1}^{n} e_{2j}x_j \right). \tag{1.7}
\]

We can continue on, and eventually reach for all \( 1 \leq k \leq l \):

\[
x_{n_k} = \frac{1}{e_{kn_k}} \left( d_k - \sum_{j=n_k+1}^{n} e_{kj}x_j \right). \tag{1.8}
\]

Hence the solution to the least squares problem (1.4) can be computed as follows:

1. Choose \( x_i, i \notin \{n_1, \ldots, n_l\} \) arbitrarily (for example, zero).

2. Use (1.8) to compute \( x_{n_l}, x_{n_{l-1}}, \ldots, x_{n_1} \) recursively.

Meanwhile, we reduce the problem to find a factorization \( A = QE \) such that \( Q \) is orthogonal and \( E \) is of the row echelon form.
2 Givens Rotation

A basic tool to find the factorization $A = QE$ is to use Givens rotations. Let us consider a simple example in $\mathbb{R}^2$:

![Givens Rotation Diagram](image)

Figure 1: Rotation by $\theta$ in $\mathbb{R}^2$.

Particularly, we want to rotate a vector $\mathbf{x} = [x, y]^t$ by an angle $\theta$ counter-clockwise to a new vector $G\mathbf{x} = [x', y']^t$. According to Figure 1, we assume:

$$x = r \cos \alpha, \quad y = r \sin \alpha,$$

where $r = ||\mathbf{x}|| = ||G\mathbf{x}||$. Then the two coordinates of $G\mathbf{x}$ are given by:

$$x' = r \cos(\alpha + \theta) = r (\cos \alpha \cos \theta - \sin \alpha \sin \theta) = \cos \theta x - \sin \theta y,$$

$$y' = r \sin(\alpha + \theta) = r (\sin \alpha \cos \theta + \cos \alpha \sin \theta) = \cos \theta y + \sin \theta x.$$

Thus we conclude that the rotation matrix $G$ is defined:

$$G = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}. \quad (2.1)$$

In multiple dimensions, we consider the rotations that keep all but two coordinates constant. In $\mathbb{R}^3$, these operations are those rotate about one of the three axes. Particularly, let the indices for the two modified coordinates be $i$ and $j$, then the rotation by an angle $\theta$ is equivalent to pre-multiplication with the Givens matrix $G_{i,j}(\theta)$.

$$G_{i,j}(\theta) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cos(\theta) & \cdots & -\sin(\theta) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & \sin(\theta) & \cdots & \cos(\theta) & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}. \quad (2.2)$$

All Givens matrices are orthogonal.
3 Orthogonal Reduction by Givens Rotations

The idea here is to apply a sequence of Givens rotations to the left of $A$ so that the latter is transformed into the row echelon form. We’ve learned from the process of Gaussian elimination that left multiplication indicates row manipulations; and we see more familiarities between the orthogonal reduction procedure here and the Gaussian elimination. That is, the last elements of a column of $A$ are transformed to zeroes by row operations. The tool of choice is lower-triangular matrices for the Gaussian elimination, whereas it is orthogonal matrices (or more specifically the product of a sequence of Givens matrices) in the current situation.

First we look at the product $G_{1,2}(\theta)A$, where $\theta$ is a number to be determined. Denote the $i$-th row of $A$ by $a_i^t$, $1 \leq i \leq m$, and we denote the $i$-th column of a generic matrix $M$ by $[M]_i$, then:

$$
G_{1,2}(\theta)A = \begin{bmatrix}
cos\theta a_1^t - \sin\theta a_2^t \\
\sin\theta a_1^t + \cos\theta a_2^t \\
a_3^t \\
\vdots \\
a_m^t
\end{bmatrix} \Rightarrow [G_{1,2}(\theta)A]_1 = \begin{bmatrix}
cos\theta a_{11} - \sin\theta a_{21} \\
\sin\theta a_{11} + \cos\theta a_{21} \\
a_{31} \\
\vdots \\
a_{m1}
\end{bmatrix}.
$$

Note that all the rows except for the first two ones are not changed at all. We may choose $\theta$ such that $\sin\theta a_{11} + \cos\theta a_{21} = 0$, or equivalently:

$$
\theta = \arctan\left(-\frac{a_{21}}{a_{11}}\right),
$$

and the $(2,1)$-element of $G_{1,2}(\theta)A$ becomes zero. The advantage of the Givens transformation over the Gaussian elimination is that (3.1) is well-defined even when $a_{11} = 0$, in which case $\theta = \pi/2$ and $G_{1,2}(\theta)$ can still be computed. We shall denote this particular Givens matrix by $G^{(1)}_{1,2}$.

Another fact we notice after the rotation is that the $L^2$-norm of the first column of $A$ is not changed. Particularly, note that if $a_{11}^2 + a_{21}^2 \neq 0$, there is:

$$
\sin\theta = -\frac{a_{21}}{\sqrt{a_{11}^2 + a_{21}^2}}, \quad \cos\theta = \frac{a_{11}}{\sqrt{a_{11}^2 + a_{21}^2}};
$$

and we have:

$$
[A]_1 \mapsto [G^{(1)}_{1,2}A]_1 \quad \text{is given by} \quad \begin{bmatrix}
a_{11} \\
a_{21} \\
a_{31} \\
\vdots \\
a_{m1}
\end{bmatrix} \mapsto \begin{bmatrix}
a_{11} \\
\sqrt{a_{11}^2 + a_{21}^2} \\
a_{31} \\
\vdots \\
a_{m1}
\end{bmatrix}.
$$

It is easy to check that in the special situation $a_{11}^2 + a_{21}^2 = 0$, the previous statement remains true.

Preserving the $L^2$-norm of the first column vector is actually true for all $\theta$ (and can be derived from the $L^2$-norm preserving property of any orthogonal matrix); and particularly we see that the
$L^2$-norm of all the column vectors of $A$ remain the same after $A \rightarrow G_{1,2}^{(1)} A$.

Next, we construct a Givens matrix $G_{1,3}^{(1)}$ that will make the (3,1)-element of $G_{1,2}^{(1)} A$ zero:

\[
[A]_1 \mapsto [G_{1,3}^{(1)} \cdot G_{1,2}^{(1)}]_1 \quad \text{is given by}
\begin{bmatrix}
    a_{11} \\
    a_{21} \\
    a_{31} \\
    a_{41} \\
    \vdots \\
    a_{m1}
\end{bmatrix}
\mapsto
\begin{bmatrix}
    a_{11} \\
    a_{21} \\
    a_{31} \\
    a_{41} \\
    \vdots \\
    a_{m1}
\end{bmatrix}
\begin{bmatrix}
    \sqrt{a_{11}^2 + a_{21}^2 + a_{31}^2} \\
    0 \\
    0 \\
    a_{41} \\
    \vdots \\
    a_{m1}
\end{bmatrix}.
\]

The matrix $G_{1,3}^{(1)}$ is given by:

\[G_{1,3}^{(1)} = G_{1,3}(\theta), \quad \text{where } \theta = \arctan \left( -\frac{a_{31}}{\sqrt{a_{11}^2 + a_{21}^2}} \right).\]

As we continue, all the remaining non-zeroes in the first column of $A$ can be eliminated. Eventually we obtain a sequence of Givens matrices and define their product as $G_1$:

\[G_1 = G_{1,m}^{(1)} G_{1,m-1}^{(1)} \cdots G_{1,2}^{(1)}, \quad (3.2)\]

so that:

\[
[A]_1 \mapsto [G_1 A]_1 \quad \text{is given by}
\begin{bmatrix}
    a_{11} \\
    a_{21} \\
    \vdots \\
    a_{m1}
\end{bmatrix}
\mapsto
\begin{bmatrix}
    a_{11} \\
    a_{21} \\
    \vdots \\
    a_{m1}
\end{bmatrix}
\begin{bmatrix}
    \sqrt{a_{11}^2 + a_{21}^2 + \cdots + a_{m1}^2} \\
    0 \\
    \vdots \\
    0
\end{bmatrix}.
\]

Let us denote $A^{(1)} = G_1 A$, then the first column of $A^{(1)}$ is exactly what we want for $E$; and if $a_{11}^{(1)} = \sqrt{a_{11}^2 + \cdots + a_{m1}^2} \neq 0$, we have $n_1 = 1$.

The next step is to use Givens rotations to eliminate as many non-zeroes elements of the second column of $A^{(1)}$ as possible. If $a_{11}^{(1)} = 0$, this process is the same as what we did before for the first column of $A$; but if $a_{11}^{(1)} \neq 0$, we want to leave the first row of $A^{(1)}$ untouched! Particularly, we construct a sequence of Givens matrices and define $G_2$ as their products:

\[
G_2 = \begin{cases}
G_{2,m}^{(2)} G_{2,m-1}^{(2)} \cdots G_{2,3}^{(2)}, & \text{if } a_{11}^{(1)} \neq 0; \\
G_{1,m}^{(2)} G_{1,m-1}^{(2)} \cdots G_{1,2}^{(2)}, & \text{if } a_{11}^{(1)} = 0.
\end{cases} \quad (3.3)
\]
such that:

\[ [A^{(1)}]_2 \rightarrow [G_2 A^{(1)}]_2 \] is given by

\[
\begin{bmatrix}
  a_{12}^{(1)} \\
  a_{12}^{(2)} \\
  a_{22}^{(1)} \\
  a_{22}^{(2)} \\
  \vdots \\
  a_{m2}^{(m)}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
  a_{21}^{(1)} \\
  \sqrt{(a_{22}^{(1)})^2 + (a_{32}^{(1)})^2 + \cdots + (a_{m2}^{(m)})^2} \\
  0 \\
  \vdots \\
  0
\end{bmatrix}, \quad \text{if } a_{11}^{(1)} \neq 0;
\]

or

\[
\begin{bmatrix}
  a_{12}^{(1)} \\
  a_{12}^{(2)} \\
  a_{22}^{(1)} \\
  a_{22}^{(2)} \\
  \vdots \\
  a_{m2}^{(m)}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
  \sqrt{(a_{21}^{(1)})^2 + (a_{22}^{(1)})^2 + \cdots + (a_{m2}^{(m)})^2} \\
  0 \\
  \vdots \\
  0
\end{bmatrix}, \quad \text{if } a_{11}^{(1)} = 0.
\]

Continuing this process, we obtain orthogonal \( m \times m \) matrices \( G_1, G_2, \ldots, G_n \) such that:

\[ G_n G_{n-1} \cdots G_1 A = E, \quad (3.4) \]

where \( E \) is of the row echelon form. Defining \( Q = (G_n G_{n-1} \cdots G_1)^t \) we obtain the desired factorization \( A = QE \).

Now we write down the algorithm rigorously in Algorithm 3.1. Here we use an integer \( p \) to keep track of the row number, below which the non-zero entries are transformed to zero.

**Algorithm 3.1 Orthogonal Reduction by Givens Rotations**

1: Set \( p = 1 \) and \( Q = I_m \)
2: for \( i = 1, 2, \ldots, n \) do
3: \hspace{1em} for \( j = p + 1, p + 2, \ldots, m \) do
4: \hspace{2em} if \( a_{ji} = 0 \) then
5: \hspace{3em} Continue
6: \hspace{2em} end if
7: \hspace{2em} Compute \( \theta = \arctan(-a_{ji}/a_{pi}) \)
8: \hspace{2em} Compute \( A \leftarrow G_{p,j}(\theta) A \)
9: \hspace{2em} Compute \( Q \leftarrow Q G_{p,j}(\theta)^t \)
10: \hspace{1em} end for
11: \hspace{1em} if \( a_{pi} \neq 0 \) then
12: \hspace{2em} Set \( p \leftarrow p + 1 \)
13: \hspace{1em} end if
14: end for

At the end of the algorithm, the matrix \( A \) is transformed into the row echelon form \( E \). Note that in the line 5, we do not have to actually compute \( \theta \) from line 4 and form the matrix \( G_{p,j}(\theta) \) but instead compute and store:

\[
c_{pj} = \frac{a_{pi}}{\sqrt{a_{ji}^2 + a_{pi}^2}}, \quad s_{pj} = -\frac{a_{ji}}{\sqrt{a_{ji}^2 + a_{pi}^2}};
\]
and then compute for \( A \):

\[
\begin{align*}
& a_{jk} \leftarrow c_{pj}a_{jk} - s_{pj}a_{pk}, \\
& a_{pk} \leftarrow s_{pj}a_{jk} + c_{pj}a_{jk}, \\
& k = i, i+1, \ldots, n.
\end{align*}
\] (3.5)

Similarly for the line 6, if the matrix \( Q \) is not explicitly needed immediately, all we need to do is to keep track of all pairs \( c_{pj} \) and \( s_{pj} \) so that \( Q \) can be reconstructed later.

### 4 Analysis of Algorithm 3.1

The preceding factorization is more robust than the Gaussian elimination because we can obtain an \textit{a priori} estimate on all the components that may appear during the orthogonalization process. In particular, whenever we apply the Givens rotation, the \( L^2 \)-norm of the column vectors are not changed; hence we have:

\[
\sum_{j=1}^{m} e_{ji}^2 = \sum_{j=1}^{m} a_{ji}^2 \implies |e_{ki}| \leq \sqrt{\sum_{j=1}^{m} a_{ji}^2},
\]

for all \( i = 1, \ldots, n \) and \( k = 1, \ldots, m \).

Next, we study the complexity of Algorithm 3.1. Note that in the outer loop, no computation actually takes place if \( p \) is not increased at all. Thus the maximum possible computational cost includes \( r = \min(m, n) \) inner loops, which correspond to the value of \( p \) as \( p = 1, p = 2, \ldots, p = r \), respectively. For a given such \( p \), the inner loop has \( m - p \) iterations. Each iteration contains (we take the approach without computing \( \theta \) explicitly) five flops and one square root operation to compute \( c_{pj} \) and \( s_{pj} \). The operations (3.5) are thusly completed with \( 6(n - i + 1) \) flops. Note that we always have \( i \geq p \), the total number of flops is thusly bounded as:

\[
\sum_{p=1}^{r} \sum_{j=p+1}^{m} (5 + 6(n - i + 1)) \leq \sum_{p=1}^{r} \sum_{j=p+1}^{m} (5 + 6(n - p + 1)) \leq \sum_{p=1}^{r} [6(n - p)(m - p) + 11(m - p)] \sim 3mr(n - r) + 3nr(m - r) + 2r^3.
\]

Finally, we improve the algorithm 3.1 in computer science considerations. Looking at each outer loop, say the first one, we start to work on the row 1 and row 2, then on the row 1 and row 3, and finally move on to row 1 and row \( m \). The objective is to “rotate” all the non-zero entries of the first column of \( A \) to the first element. If we take into memory storage into account, it is usual practice to store the elements of a matrix \( A \) row by row (this can be true for both full matrices and sparse matrices); hence we’re motivated to operate on adjacent rows as often as possible in order to improve the bandwidth usage and reduce cache misses. Such a consideration results in a “roll-back” algorithm to eliminate the non-zeros – we first work on the last two rows and make the \( m \)-th element zero, then Givens rotation is applied to the rows \( m - 2 \) and \( m - 1 \) to make the \((m - 1)\)-th element zero, and finally we reach the top of the column. This modification is reflected in Algorithm 4.1. Note that we also incorporate the computations of \( c \)'s and \( s \)'s instead of \( \theta \) in this modified version.
Algorithm 4.1 Orthogonal Reduction by Givens Rotations (Modified)

1: Set $p = 1$ and $Q = I_m$
2: for $i = 1, 2, \ldots, n$ do
3: \hspace{1em} for $j = m, m-1, \ldots, p+1$ do
4: \hspace{2em} if $a_{ji} = 0$ then
5: \hspace{3em} Continue
6: \hspace{2em} end if
7: \hspace{2em} Compute $c_{j-1,j} = a_{j-1,i} / \sqrt{a_{ji}^2 + a_{j-1,j}^2}$ and $s_{j-1,j} = -a_{ji} / \sqrt{a_{ji}^2 + a_{j-1,j}^2}$
8: \hspace{2em} Compute $A \leftarrow G_{j-1,j}(\theta)A$
9: \hspace{2em} Compute $Q \leftarrow QG_{j-1,j}(\theta)^t$
10: \hspace{1em} end for
11: \hspace{1em} if $a_{pi} \neq 0$ then
12: \hspace{2em} Set $p \leftarrow p + 1$
13: \hspace{1em} end if
14: end for