orthogonalization

L. Olson

October 27, 2015

Department of Computer Science
University of Illinois at Urbana-Champaign
objectives

• Revisit SVD and Orthogonal Matrices
• Create orthogonal vectors
• Outline the Gram-Schmidt algorithm for orthogonalization
The normal equations tend to worsen the condition of the matrix.

**Theorem**

\[
\text{cond}(A^T A) = (\text{cond}(A))^2
\]
other approaches

- **QR factorization.**
  - For $A \in \mathbb{R}^{m \times n}$, factor $A = QR$ where
    - $Q$ is an $m \times m$ orthogonal matrix
    - $R$ is an $m \times n$ upper triangular matrix (since $R$ is an $m \times n$ upper triangular matrix we can write $R = \begin{bmatrix} R' \\ 0 \end{bmatrix}$ where $R$ is $n \times n$ upper triangular and 0 is the $(m - n) \times n$ matrix of zeros)

- **SVD - singular value decomposition**
  - For $A \in \mathbb{R}^{m \times n}$, factor $A = USV^T$ where
    - $U$ is an $m \times m$ orthogonal matrix
    - $V$ is an $n \times n$ orthogonal matrix
    - $S$ is an $m \times n$ diagonal matrix whose elements are the singular values.
orthogonal matrices

Definition

A matrix $Q$ is orthogonal if

$$Q^T Q = QQ^T = I$$

Orthogonal matrices preserve the Euclidean norm of any vector $v$, 

$$\|Qv\|_2^2 = (Qv)^T (Qv) = v^T Q^T Qv = v^T v = \|v\|_2^2.$$
One way to obtain the \( QR \) factorization of a matrix \( A \) is by Gram-Schmidt orthogonalization.

We are looking for a set of orthogonal vectors \( q \) that span the range of \( A \).

For the simple case of 2 vectors \( \{a_1, a_2\} \), first normalize \( a_1 \) and obtain

\[
q_1 = \frac{a_1}{\|a_1\|}.
\]

Now we need \( q_2 \) such that \( q_1^T q_2 = 0 \) and \( q_2 = a_2 + cq_1 \). That is,

\[
R(q_1, q_2) = R(a_1, a_2)
\]

Enforcing orthogonality gives:

\[
q_1^T q_2 = 0 = q_1^T a_2 + cq_1^T q_1
\]
gram-schmidt orthogonalization

\[ q_1^T q_2 = 0 = q_1^T a_2 + cq_1^T q_1 \]

Solving for the constant \( c \).

\[ c = -\frac{q_1^T a_2}{q_1^T q_1} \]

reformulating \( q_2 \) gives.

\[ q_2 = a_2 - \frac{q_1^T a_2}{q_1^T q_1} q_1 \]

Adding another vector \( a_3 \) and we have for \( q_3 \),

\[ q_3 = a_3 - \frac{q_2^T a_3}{q_2^T q_2} q_2 - \frac{q_1^T a_3}{q_1^T q_1} q_1 \]

Repeating this idea for \( n \) columns gives us Gram-Schmidt orthogonalization.
Since $R$ is upper triangular and $A = QR$ we have

\[
\begin{align*}
  a_1 &= q_1 r_{11} \\
  a_2 &= q_1 r_{12} + q_2 r_{22} \\
  \vdots &= \vdots \\
  a_n &= q_1 r_{1n} + q_2 r_{2n} + \ldots + q_n r_{nn}
\end{align*}
\]

From this we see that $r_{ij} = \frac{q_i^T a_j}{q_i^T q_i}, j > i$
The orthogonal projector onto the range of $q_1$ can be written:

\[
\frac{q_1 q_1^T}{q_1^T q_1}
\]

. Application of this operator to a vector $a$ orthogonally projects $a$ onto $q_1$. If we subtract the result from $a$ we are left with a vector that is orthogonal to $q_1$.

\[
q_1^T (I - \frac{q_1 q_1^T}{q_1^T q_1}) a = 0
\]
def qr(A):
    Q = np.zeros(A.shape)
    for k in range(A.shape[1]):
        avec = A[:, k]
        q = avec
        for j in range(k):
            q = q - np.dot(avec, Q[:,j])*Q[:,j]