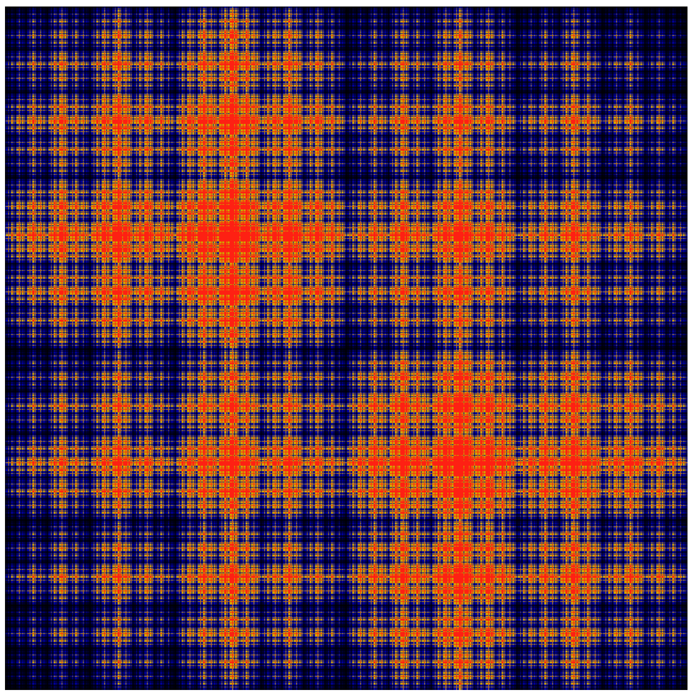
Applied Mathematics
and
Computing

Volume I



List of Contributors

J. Humpherys
Brigham Young University

J. Webb
Brigham Young University

R. Murray
Brigham Young University

J. West
University of Michigan

R. Grout
Brigham Young University

K. Finlinson
Brigham Young University

A. Zaitzeff
Brigham Young University

Preface

This lab manual is designed to accompany the textbook *Foundations of Applied Mathematics* by Dr. J. Humpherys.

Lab 9

Algorithms: Modified Gram-Schmidt (QR)

Lesson Objective: *Understand how the QR algorithm works and write your own implementation.*

The QR decomposition is used to represent any matrix as the multiple of an orthogonal matrix and an upper triangular matrix. This decomposition is useful in computing least squares and is part of a common method for finding eigenvalues.

Review of Gram Schmidt

Theorem 9.1 (Gram-Schmidt Orthogonalization Process). *Let $\{\mathbf{x}_i\}_{i=1}^n$ be a basis for the inner product space V. Let*

$$\mathbf{q}_1 = \frac{\mathbf{x}_1}{\|\mathbf{x}_1\|},$$

and define $\mathbf{q}_2, \mathbf{q}_3, \ldots, \mathbf{q}_n$ recursively by

$$\mathbf{q}_{k+1} = \mathbf{x}_{k+1} - \sum_{j=1}^{k} \frac{\langle \mathbf{x}_{k+1}, \mathbf{q}_j \rangle}{\|\mathbf{q}_j\|^2} \mathbf{q}_j,$$

the sum term is a projection of $\mathbf{x}_{k+1}$ onto the subspace Span$(\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_k)$. Then the set $\{\mathbf{q}_i\}_{i=1}^n$ is an orthonormal basis for V.

For the above algorithm, let $r_{jk} = \langle \mathbf{x}_k, \mathbf{q}_j \rangle$ when $j \leq k$. Then

$$r_{11}\mathbf{q}_1 = \mathbf{x}_1$$
$$r_{kk}\mathbf{q}_k = \mathbf{x}_k - r_{1k}\mathbf{q}_1 - r_{2k}\mathbf{q}_2 - r_{3k}\mathbf{q}_3 - \ldots - r_{k-1,k}\mathbf{q}_{k-1}, \quad k = 2, \ldots, n.$$

This can be written as

$$\begin{aligned}
\mathbf{x}_1 &= r_{11}\mathbf{q}_1 \\
\mathbf{x}_2 &= r_{12}\mathbf{q}_1 + r_{22}\mathbf{q}_2 \\
\vdots &= \quad \vdots \\
\mathbf{x}_n &= r_{1n}\mathbf{q}_1 + r_{2n}\mathbf{q}_2 + \ldots + r_{nn}\mathbf{q}_n,
\end{aligned}$$

or in matrix form as

$$\begin{pmatrix} \vdots & \vdots & & \vdots \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \\ \vdots & \vdots & & \vdots \end{pmatrix} = \begin{pmatrix} \vdots & \vdots & & \vdots \\ \mathbf{q}_1 & \mathbf{q}_2 & \cdots & \mathbf{q}_n \\ \vdots & \vdots & & \vdots \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{22} & \cdots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{nn} \end{pmatrix}.$$

Hence if our original basis $\{\mathbf{x}_i\}_{i=1}^n$ correspond to column vectors of a matrix A, we can likewise write the resulting orthonormal basis $\{\mathbf{q}_i\}_{i=1}^n$ as a matrix Q of column vectors. Then we have that $A = QR$, where R is the above nonsingular upper-triangular $n \times n$ matrix. This is the QR Decomposition and is summarized by the following theorem:

Theorem 9.2. *Let A be an $m \times n$ matrix of rank n. Then A can be factored into a product QR, where Q is an $m \times n$ matrix with orthonormal columns and R is a nonsingular $n \times n$ upper triangular matrix.*

There are three mode options available in SciPy's implementation of QR Decomposition. We will be using the "economic" option.

```
: import scipy as sp
: from scipy import linalg as la
: A = sp.randn(4,3)
: Q, R = la.qr(A, mode='economic')
: sp.dot(Q, R) == A      there will be some False entries
: sp.dot(Q, R) - A
: sp.dot(Q.T, Q)
```

In order to interpret the results correctly, we need to understand that the computer has limited precision (especially with floating point numbers). This is why `sp.dot(Q, R)` is not exactly equal to A. But subtracting the two yields numbers that are essentially zero. This shows that indeed the product of Q and R is A. Note also that $Q^T Q = I$. This implies that the column vectors of Q are orthonormal (why?).

Solving Least Squares Problems

For large or ill-conditioned problems, the QR decomposition provides a nice method for computing least squares solutions of over-determined matrices. Consider the problem $Ax = b$. Recall that the least squares solution is $\widehat{x} = (A^TA)^{-1}A^Tb$. Alternatively, we write the linear system as

$$QRx = b.$$

We then multiply both sides by Q^T, yielding

$$Rx = Q^Tb.$$

Then $\widehat{x} = R^{-1}Q^Tb$.

Computational Remark

Numerically, the Gram Schmidt process can have problems due to finite precision arithmetic. Specifically, due to rounding errors, the resulting basis may not be orthonormal. To combat this, we actually carry out a slightly revised algorithm called Modified Gram Schmidt. To do this, we compute $\mathbf{q}_1$ as before. We then project it out of each of the remaining original vectors $\mathbf{x}_2, \mathbf{x}_3, \ldots, \mathbf{x}_n$ via

$$\mathbf{x}_k := \mathbf{x}_k - \langle \mathbf{x}_k, \mathbf{q}_1 \rangle \mathbf{q}_1, \quad k = 2, \ldots, n.$$

Then we compute $\mathbf{q}_2$ to be the unit vector of $\mathbf{x}_2$, that is,

$$\mathbf{q}_2 = \frac{\mathbf{x}_2}{\|\mathbf{x}_2\|}.$$

We repeat by projecting out $\mathbf{q}_2$ from the remaining vectors $\mathbf{x}_3, \mathbf{x}_4, \ldots, \mathbf{x}_n$.

Problem 1 Write your own implementation of the QR decomposition. It should accept as input a matrix A and computes its QR decomposition, returning the matrices Q and R. Be sure to use the numerically stable Modified Gram Schmidt algorithm.