The gram-schmidt process is a method for orthonormalizing a set of vectors in an inner product space. Most commonly, the Euclidean space, R^n, equipped with the standard inner product.

Task A:

Line 11: The input matrix A is defined

Line 13: [U,D] = eig(A) returns a diagonalized matrix, D, of eigenvalues and a matrix, U, whose columns are the corresponding right eigenvectors such that U*D = A*U.

Line 14: evals = diag(D) stores the diagonal elements of D into a vector called evals. This works due to D being a diagonal matrix with the eigenvalues of A stored on the main diagonal.

Line 22: [dummy, ind] = sort(abs(evals), “descend”) This line sorts the magnitude of the eigenvalues from largest to smallest, with ‘dummy’ storing these values, and ‘ind’ storing the index of the value from the original vector. Eg [3, -5] gets sorted to [5, 3] with ind storing [2, 1] stating that the second element from the original eigenvalue vector is now stored in the first position.

Line 23: evals = evals(ind). This line reorders the original eigenvalue vector based on the indexes returned from the above reordering.

Line 24: U = U(:, ind). This line reorders each column in the matrix U according to the indexes found earlier.

Line 47: U(:,[1:4]) = -U(:,[1:4]). This line multiplies some eigenvectors by a negative as the code is only focusing on positive variables. This is permissible because eigenvectors multiplied by a scalar are still eigenvectors.

Line 55: U(:,1)’ * U(:,3) . This line prints the overlap between the first and third column of the matrix U. As the resulting scalar is 0.3015, it indicates the vectors are not orthogonal

Line 59: U’*U. This line prints the overlap between all of the vectors as a matrix.

Gram-Schmidt Process

Line 80: V(:,1) = U(:,1). The first vector in the orthogonal space is given the value of the first column of U.

Line 86-88: V(:,1)’ *U(:,2), V(:,2) = U(:,2). The overlap between the first and second column of U. In this case, there is no overlap. This can also be seen from the resulting matrix of U’*U where the value at row 1, column 2 is 0. As such, the second column of V can be the same as the second column of U.

Lines 93 & 94: V(:,3) = U(:,3) – (U(:,3)’*V(:,1)) * V(:,1), V(:,3) = V(:,3) / norm(V(:,3)). The overlap between the first column of V and the third column of U is calculated and subtracted from the third column of U. This result in a vector that is orthogonal to both the first and second column of V. This vector is then normalized, resulting in a vector with a length of 1.

Line 107: V’*V. This line prints the result of multiply V transpose by V. As the result is the identity matrix, it confirms that the matrix is orthogonal and that the vectors inside have unit length.

Lines 113 and 114. The overlap of the fourth column of U with the other columns already in V is calculated ad subtracted. The resulting orthogonal vector is then normalized and stored as the 4th column of V.

Line 120: V'*V. This line again prints the result of multiply V transpose by V. As the result is the identity matrix, now with 4 rows and 4 columns, it confirms that the matrix is orthogonal and that the vectors inside have unit length.

Line 19 of gramSchmidt_errorcheck.m shows that if any one of the vectors in U has a magnitude of 0, then the vectors inside are not linearly independent.