The purpose of this assignment was to get a better understanding of eigen values, covariance, and least squares optimization of linear regression with and without regularization at varying degrees of polynomial regression.

The covariance matrix is the dot product of the mean-centered x and y datapoints divided by n-1 observations. The covariance matrix is a symmetric matrix that contains the variance in the x and y directions (along the diagonal of the covariance matrix) as well as any diagonal correlations/orientation of the data (along the non-diagonals of the covariance matrix). These four variance/covariance values make up the covariance matrix, for 2-D data it will be a 2x2 matrix. For this assignment the covariance matrix was determined manually using the instructions listed above, but can also be calculated using numpy.cov().

Eigen decomposition can be taken of the covariance matrix, yielding a transformation matrix that transforms any data along the variance and orientation of the data we observe. For 2-D data, we get two eigenvectors, the largest of which points in the direction of the largest variance (most spread). The second largest eigenvector is orthogonal to the first and corresponds to the axis of variance with distances greatest from the first axis. The purpose of eigen decomposition is to reduce the number of dimensions associated with a data space, allowing us to make regressions for modeling purposes. For this assignment, the eigenvalues and eigenvectors were calculated using the numpy.linalg.eig() function.

When performing least squares line fitting to the datasets I created columns of 1's for x0, the actual x and y data for x1 and y1, and then columns for x2 (x squared) and x3 (x cubed) for a degree 3 polynomial regression line model. The 4 theta values for the x0, x1, x2, x3 parameters were obtained by taking the dot product between the inverse gram matrix (X.T @ X)^-1 and the moment matrix (X.T @ Y). This line fitting was performed for all four datasets. With a degree 3 multivariate regression the fitting was not overwhelmingly close except for dataset 4 which may have been too biased due to the higher dimensionality.

For regularization with ridge regression, the same x0, x1, x2, x3 columns were used. The theta values in ridge regression were obtained first by mean centering, followed by multiplying an identity matrix by either a small lambda (<5) or large lambda (>100) and adding that to (X.T @ X) before taking the dot product between the inversed gram matrix and moment matrix.

In general, the higher the lambda input that was used, the more bias was introduced. Indeed, for high lambda values of 100,000,000 in Dataset1 for example, a fair amount of bias was introduced into the line model which could more accurately be modeled with two dimensions. The result of ridge regularization with a large lambda made the model more sensitive to noise than with a smaller lambda value. This trend became apparent for all datasets, some needing extremely large lambda values before the line model became unusable as the higher degree terms began losing their magnitude. It is my interpretation that dataset4 had the smallest amount of variance and outliers, making the regularized line fitting not very accurate. With so little variance in dataset4, any lambda value big or small seemed to introduce too much bias to make it a good model predictor in this case.

Some limitations of ridge regression include the fact that it introduces bias depending on the eigenvalues used, causing a bias/variance tradeoff. Another limitation is that ridge regression is degree reduction method and can become very taxing to compute with a high dimensionality. And finally, for datasets with already low variance, ridge regression can actually worsen line fitting performance.