1. The reason for using the vector container rather than any other available in the STL.

In this workshop, I don’t need to add or remove the element from the begin or the middle of the container. As a result, we can simply choose the vector to fulfill all the requirements in this workshop.

• List the STL template functions that you used in your solution.

Accumulate() to calculate the mean, sigma, slope and intercept.

Sort() to sort the container in order to find the median.

inner\_product() to calculate the slope and intercept.

• Identify where you used lambda expressions.

When calculate the sigma, I specific how it accumulates the data.

std::accumulate(this->dataY.begin(), this->dataY.end(), (T)0, [=](T x, T y) {

return x + pow((y - mean), 2);

});

When calculate the slope, I specific how it accumulates the data.

auto sumXX = std::accumulate(this->dataX.begin(), this->dataX.end(), (T)0, [](T x, T y) {return x + y \* y; });

• Comment on the ease of programming associated with the STL.

STL provides many common template classes to help the programmer develop efficiently. For example, with the STL container vector, we can easily handle the size of the array. With the push\_back(), we can quickly add the element into the container. We don’t need to allocate then delete the memory for array by ourselves anymore. Moreover, there are many built-in Algorithms such as sort() to decrease the development time. We can even use the lambda function in those STL functions to change the way it works. As a result, STL really helps the programmer a lot.