OSU Mechanical Engineering

# Smart Products Laboratory

Software I | One

ME Course Number
Spring 2019

# Table of Contents

# 1 Overview

In this laboratory, you will be programming vector and matrix functions in C++ to allow you to either refresh or strengthen your programming skillset. An example file set will be provided in this lab so you have a reference to go off of, however you are strongly encouraged to write your own code, as this is the only way you will learn programming.

# 2 Background

None needed for this lab.

# 3 References & Resources

## 3.1 Readings

Refer to the textbooks suggested in the syllabus for general programming questions. Additionally, there is an article on matrix inversion in Carmen. Please read through the entire lab prior to starting work on the exercises.

## 3.2 Resources

The following websites may be useful for this lab.

1. https://en.cppreference.com
2. https://en.cppreference.com/w/cpp/container/vector

# 4 Pre-lab

## 4.1 Hardware

A computer and your raspberry pi.

## 4.2 Software

Be sure you have your RPi setup, updated, and a working text editor installed.

# 5 Laboratory

Complete the exercises and save all code. You will combine the code for this lab with next week's lab and submit your work with your solutions to questions in a single submission with a single report following Lab Two's completion.

## 5.1 Procedure

1. Download the example code on Carmen.
2. Follow along as we go over a few examples in the script, but feel free to make your own files and code.
3. To compile the example code, use the command
    *b. g++ -o labone labone.cpp SPmath.h SPmath.cpp*

## 5.2 Exercises

### *5.2.a One: matrix multiplication functions*

Using a multi-file layout (i.e. FileName.h, FileName.cpp), create a matrix multiplication function for any two matrices (A, B) whose dimensions are $A \in R^{m \times n}$ and $B \in R^{n \times d}$. In other words,

$$\mathrm{C} = \mathrm{A} \times \mathrm{B}$$

C will have m rows and d columns, A will have m rows and n columns, and B will have n rows and d columns. Be sure that you enable runtime error checks for dimension mismatch. Test your algorithm for various combinations of (m,n,d) for instance,

1. $m = 4\,, n = 4, d = 6$
2. $m = 9, n = 7, d = 9$
3. $m = 6, n = 3, d = 2$
4. $m = 1, n = 1, d = 1$

### *5.2.b Two: Matrix Manipulation*

Using C++, develop the following functions, being sure to include runtime error checks for solution existence and dimensional conditions, be sure to demonstrate various test cases (i.e. test for solution existence for the inverse of a matrix)

1.) A function that computes the inverse of a matrix,
    a. create two versions of the inverse function, one that simply returns the inverse of the input matrix and another that takes a reference input matrix and assigns the inverse to it, i.e.
        i. MatF_t inv1(MatF_t mat_a);
        ii. void inv2(MatF_t& mat_a) ;
2.) a function that computes and returns the transpose of a matrix (see example in provided code).

(Reference MATLAB functions, inv() and transpose() to compare your algorithms' numerical answers. See handout in carmen for review of how to invert a matrix.)