MSIM 441/541 & ECE 406/506

Computer Graphics & Visualization

Programming Assignment One

Input Analysis Using Histograms

Thomas J Laverghetta

Computational Modeling and Simulation Engineering

Tlave002@odu.edu

757.620.7607

Virtual

10/21/20

# Introduction

# Program Design

The implementation design documentation for the programming assignment was made using Doxygen. Doxygen is a tool for generating documentation from annotated C++ sources (classes, enums, variables, functions, etc.), and programming languages (C, Objective-C, Java, etc.). To make documentation with Doxygen you comment your source code with specified markers and description of the code. The markers give specific meaning to the comments. These markers can be found on their website (<https://www.doxygen.nl/manual/docblocks.html>). After commenting my code, I generated HTML documentation (Assignment\_1\_Laverghetta\_Thomas.chm). The program documentation illustrates the functions and variables used. The functions and variables will all have short briefing explaining their use within the program and any corresponding dependencies. Doxygen will also illustrate how the functions are connected and where dependencies lay.

# Results

A user-interface (UI) menu allows users to choose what data files to input to produce histogram, what theorical distribution to augment over histogram, what number of bins for histogram, and what parameter step size. With these inputs, the user can interact with the program to meet his/her data analysis requirements.

In this section, I will describe what tasks were done to achieve UI response requirements. In total there are five-tasks: reading input data, probability density histogram, probability density functions, input analysis, and visual perception.

## Reading Input Data

The program reads data from input files to generate histogram. After a user has chosen what data file to run using user-interface menu, the program will get data file and start reading it. The data files are ASCII formatted. The first line of data files contains the number of data points in the file and each subsequent line contains one data point. Using this knowledge, the program will read and save the data. While saving the data, it will determine the maximum and minimum data points. The maximum and minimum will be used to find bin step sizes for the histogram.

## Probability Density Histogram

After reading in data points, the program will start generating histogram values. The values it will be generating are bin step-size, location of each bin endpoint, probabilities for each bin, and density of probability maximum. These values are generated using data inputted and parsed and number of intervals (bins) chosen by user (30-, 40-, and 50-bins).

Bin step-size is calculated by taking the maximum and minimum datapoints difference divided by number of intervals. i.e., the distance of datapoints then segmenting by number of intervals.

Location of each bin endpoint is calculated by additively iterating step-by-step to each bin and getting its endpoint. In pseudocode:

*Endpoints (step-size, number of intervals)*

*1. create endpoint array with number of intervals + 1 elements*

*2. endpoint [0] = minimum data point*

*3. for j in range(number of intervals):*

*4. endpoints [j] = minimum + j \* step-size*

*5. end for*

*6. return endpoints*

The probability of each bin is calculated by taking the counting the number of points within a bin then dividing by total number of points and bin step size. This will produce the density of probability of that bin (area of bin is the probability of points occurring). The density of probability maximum can be found be finding the maximum value produced in this process.

## Probability Density Function

The user can choose between two different probability density functions (distribution functions), normal and exponential distributions. The normal and exponential distribution functions is shown in equation (eqn) 1 and 2, respectively. These equations are used compute theorical curves given respective distribution parameters () and number of curve points (100 for this program). The distribution parameters can be altered during program execution using keyboard input. The parameter decrease and increase with Left and Right arrow keys, respectively, and parameters and increase and decrease with Up and Down keys, respectively. All increasing and decreasing amounts are the same, parameter step-size. Parameter step-size is chosen by user via UI (0.01-, 0.02-, and 0.05-parameter step-sizes). Once the theorical curves have been computed, they will be augmented over histogram (illustrated in Figure 1).

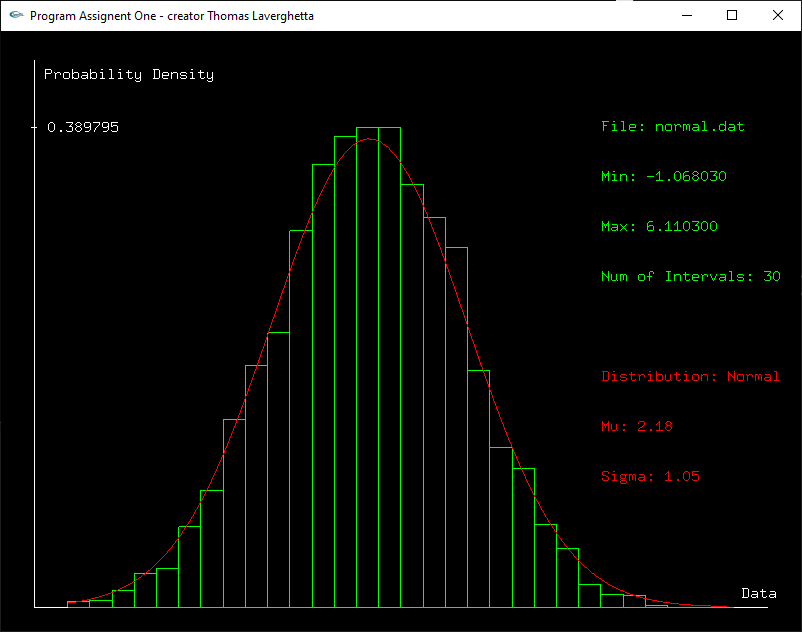


Figure . Theorical Curve Augmented on Histogram

## Input Analysis

To test my program, I used two data files with unknown theorical distributions. The data files used were data files 5.dat and 15.dat (corresponding to my first and last initials – T, L). These data files were inputted into the system and I used visual comparison to determine the best fitting theorical distribution. Figure 2 and Figure 3 show outputted histogram and the best theorical distribution ( for Figure 2 and for Figure 3).

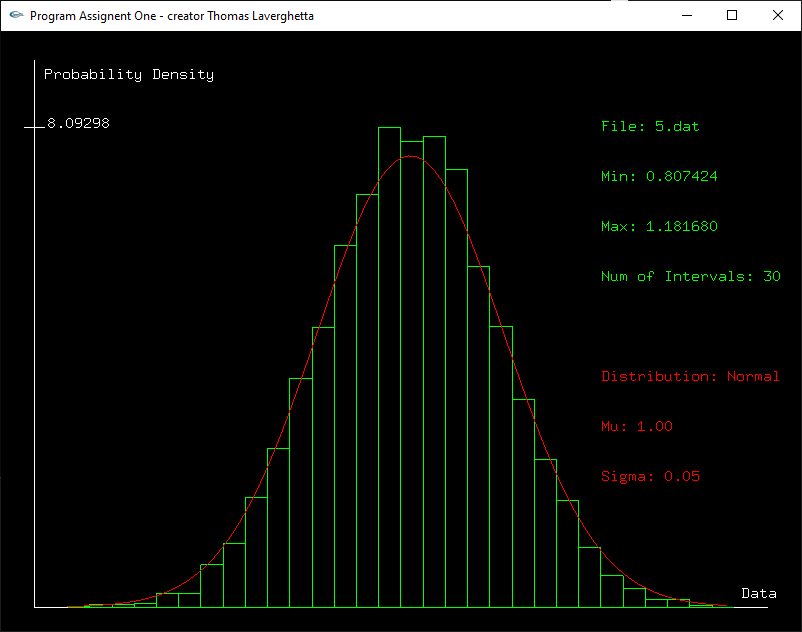


Figure . Data file 5 (5.dat) with best fitness ()

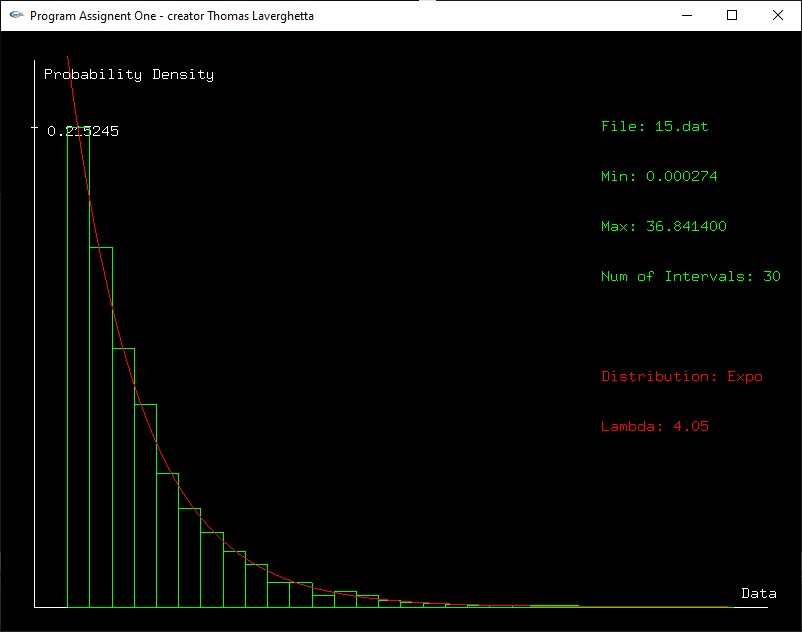


Figure . Data file 15 (15.dat) with best fitness ()

## Visual Perception

For the last task, I tried different color combinations to choose the best combination for displaying information. For this task, I tried three different color combinations as shown in Figure 4, Figure 5, and Figure 6. In the first figure, I used a black background with green to display data information (histogram, file info, min and max, and number of interval used)

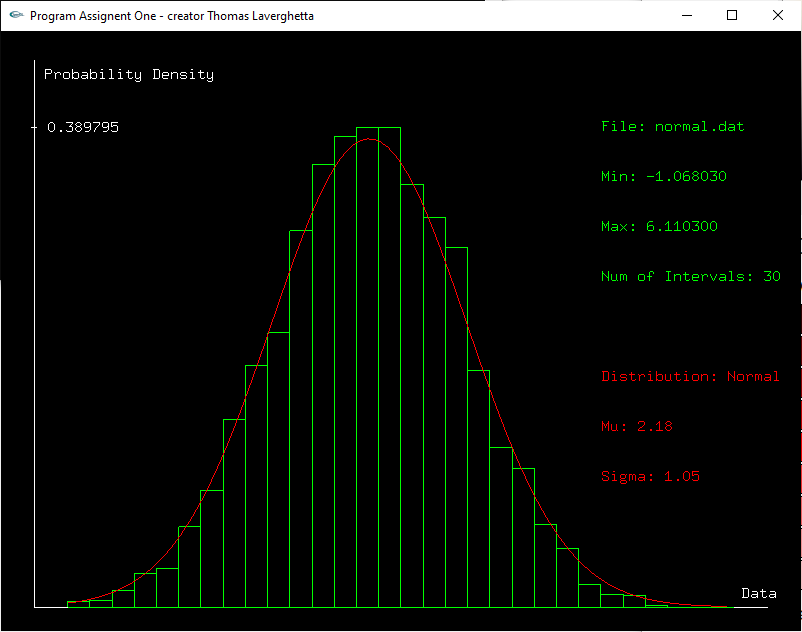


Figure . Black Background, Green Data Info, White Axis Info, and Red Distribution Info

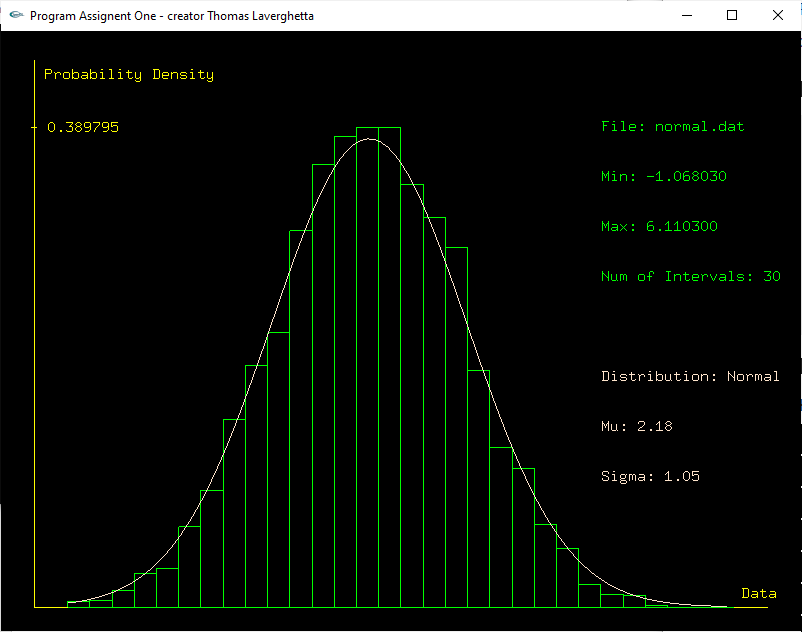


Figure . Black Background, Green Data Info, Yellow Axis Info, and Tan Distribution Info

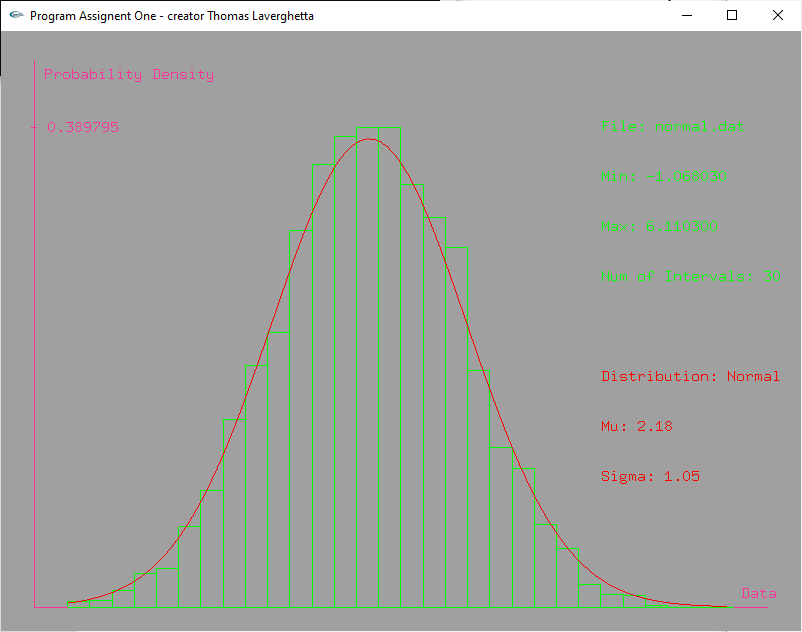


Figure . Grey background, Green Data Info Font, Pink Axis Info, and Red Distribution Info

# Conclusion and Discussion