BIEN 3200: Lab 2

**Descriptive Statistics Using Arrays, Looping, File I/O Functions, and Modular Programming**

Peter Dobbs

Lab Section 401

Nicholas Heugel

27 September 2016

**Abstract:**

**Introduction:**

The use of arrays, looping, and file input/output (I/O) functions to perform descriptive statistics on a given set of data is introduced in this lab exercise. Modular program design is implemented to execute the statistical operations. MatLab is linked to from C code to produce the desired graphics. These capabilities are all important for signal processing as it relates to biomedical engineering because these operations will become necessary during data analysis in research and product development.

**Methods:**

Equations:

(1)

(2)

(3)

(4)

(5)

Data Files:

Physiologic\_data.xls (1)

This spreadsheet contains physiologic data from over a dozen normal patients. The systolic pressure and diastolic pressure from the first ten subjects are used in the program for part 1.

SYS\_P.txt (2)

This text file contains resting systolic arterial pressures for a normal sample group.

HR.txt (3)

This text file contains resting heart rate values for a normal sample group.

sinus2.txt (4)

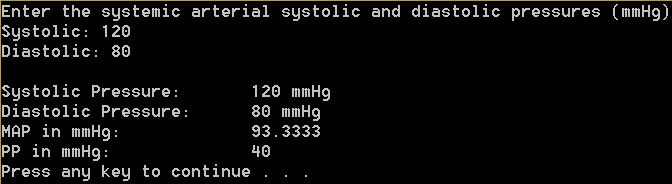
This text file contains normal sinus rhythms and the timing for the rhythms for a given sample group.

The program for part 1 of this lab called for a function that calculates the mean arterial pressure (MAP) and pulse pressure (PP) on an input systemic arterial systolic and diastolic pressures for a given subject. The program used equations 1 and 2 to accomplish the task of calculating the MAP and PP, respectively. These equations are each run in separate functions, which the main function calls, in order to perform the operations on the input values.

The program for part 2 of this lab requires a function that calculates the mean, variance, and skewness on an input dataset. It writes the results to a text file. Then a connection to MatLab is established and a histogram for the dataset is produced.

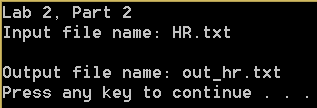
For part 3 of this lab, a program utilizes a connection to MatLab to produce a plot of a patient’s normal sinus rhythm. The dataset from sinus2.txt (data file 4) is necessary for this program. MatLab should plot the data as a function of amplitude (mV) versus time (seconds).

**Results:**

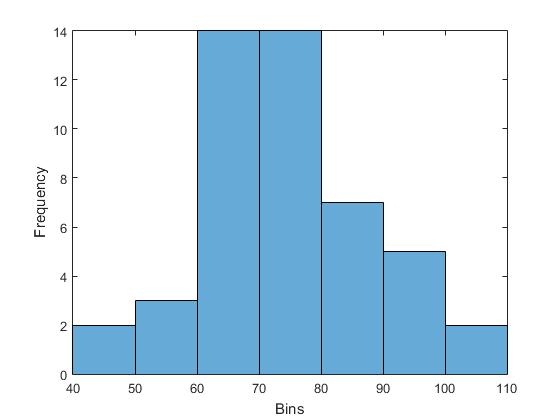


**Figure 1:** Screenshot of the output of MAP and PP functions from the program for part 1

|  |  |  |  |
| --- | --- | --- | --- |
| **Systolic Pressure** | **Diastolic Pressure** | **Mean Arterial Pressure** | **Pulse Pressure** |
| 128 | 77 | 94 | 51 |
| 108 | 74 | 85.3 | 34 |
| 138 | 76 | 96.6 | 62 |
| 165 | 91 | 115.6 | 74 |
| 140 | 64 | 89.3 | 76 |
| 132 | 72 | 92 | 60 |
| 154 | 74 | 100.6 | 80 |
| 131 | 66 | 87.6 | 65 |
| 109 | 63 | 78.3 | 46 |
| 152 | 97 | 115.3 | 55 |



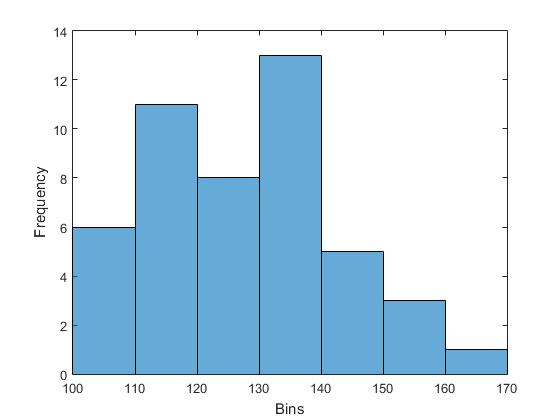
**Figure 2:** Screenshot of the console of the program from part 2



**Figure 3:** Histogram of heart rate data from HR.txt for part 2

|  |
| --- |
| mean: 73.8  variance: 0.00011  skew: -1.15e+28 |

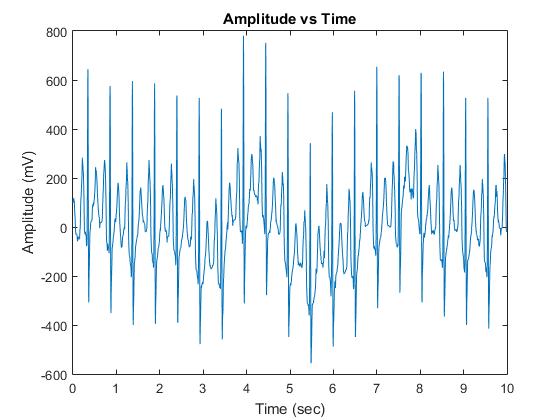
**Figure 4:** Contents of output file for part 2 program with HR.txt as input data set.



**Figure 5:** Histogram of data from SYS\_P.txt for part 2

|  |
| --- |
| mean: 127  variance: 9.11e-05  skew: -1.39e+28 |

**Figure 6:** Contents of output file for part 2 program with SYS\_P.txt as input data set.

****

**Figure 7:** Plot of normal sinus rhythm for part 3

**Discussion:**

The modularization of code that performs data analysis on a set of data was successful by following best practices as shown in examples from class. The ability of C to interface with MatLab is a very handy tool for signal processing. It will certainly be useful for future projects. Unfortunately the speed of the program is still an issue. Since the libraries have to be loaded into the program in order for it to run, each time the C code establishes a connection with MatLab, the code hits a point of idling. Hopefully this will not be too much of an issue in the future.

**Conclusion:**

The objective of implementing signal analysis using modular programming, arrays, looping, and file I/O functionality was accomplished. The three programs exhibit the correct usage, as outlined is the lab description.

**Answers to Questions:**

1. MAP, as defined by equation 1, is not the arithmetic average of the systolic and diastolic pressures. The arithmetic average of the two pressures would be the sum of the pressures divided by 2.
2. A common cause for an increase in pulse pressure is leaky valves, as often occurs in elderly patients. A common cause for a decrease in pulse pressure indicates poor heart function.

**C Code:**

/\* SimpleCProgram.c : Defines the entry point for the console application.

Peter Dobbs

BIEN 3200 - Section 401

13 September 2016

Lab 2

Program Description:

This program calculates and displays the mean arterial pressure

and the pulse pressure for a patient whose systolic and diastolic

pressures are inputs.

\*/

#include "stdafx.h"

#include <stdio.h>

float CalcMeanArterialPressure(float systolic, float diastolic) {

//Calculate the mean arterial pressure (MAP)

float MAP = diastolic + (systolic - diastolic) / 3;

return MAP;

}

float CalcPulsePressure(float systolic, float diastolic) {

//Calculate the pulse pressure (PP)

float PP = systolic - diastolic;

return PP;

}

int main(void) {

float systolic, diastolic;

//User Input

// blood pressure

printf("Enter the systemic arterial systolic and diastolic pressures (mmHg)\n");

printf("Systolic: ");

scanf\_s("%g", &systolic);

printf("Diastolic: ");

scanf\_s("%g", &diastolic);

//Calculate mean arterial pressure (MAP)

float MAP = CalcMeanArterialPressure(systolic, diastolic);

//Display systolic pressure, diastolic pressure, and MAP

printf("\nSystolic Pressure:\t %g mmHg", systolic);

printf("\nDiastolic Pressure:\t %g mmHg", diastolic);

printf("\nMAP in mmHg:\t\t %g", MAP);

//Calculate and display pulse pressure (PP)

float PP = CalcPulsePressure(systolic, diastolic);

printf("\nPP in mmHg:\t\t %g\n", PP);

system("PAUSE"); //pauses command window for user to view results

return 0;

}

/\* lab2part2.cpp : Defines the entry point for the console application.

Peter Dobbs

BIEN 3200 - Section 401

13 September 2016

Lab 2

Program Description:

This program takes in a column of float data values and performs

statistic calculations (mean, variance, skew) on it.

\*/

#include "stdafx.h"

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include "engine.h"

#include "mex.h"

#include <string.h>

/\*

data : float array that holds data set

n : int representing size of array

Calculates the mean of the input set of data

\*/

float fmean(float data[], int n) {

float mean = 0;

for (int i = 0; i < n; i++)

mean = mean + data[i];

mean = mean / n;

return mean;

}

/\*

data : float array that holds data set

n : int representing size of array

Calculates the variance of the input data set

\*/

float fvariance(float data[], int n) {

float variance = 0;

float mean = fmean(data, n);

for (int i = 0; i < n; i++)

variance = variance + pow(data[i] - mean, 2);

variance = 1 / (variance - 1);

return variance;

}

/\*

data : float array that holds data set

n : int representing size of array

Calculates the skew of the input data set

\*/

float fskewn(float data[], int n) {

float skew = 0;

float mean = fmean(data, n);

float variance = fvariance(data, n);

for (int i = 0; i < n; i++)

skew = skew + pow(data[n] - mean, 3);

skew = skew / ((n - 1)\*pow(variance, 3 / 2));

return skew;

}

int main() {

printf("Lab 2, Part 2");

float xdata[100];

static float mean, var, skewN;

static int n;

FILE \*inputfile, \*outputfile;

char inputName[10];

char outputName[10];

//prompt user for name

printf("\nInput file name: ");

scanf("%s", &inputName);

// read in data

inputfile = fopen(inputName, "r");

if (inputfile == NULL) { //file not found

printf("Input file not found!\n");

system("PAUSE");

return 1; //end program

}

for (n = 0; fscanf(inputfile, "%g", &xdata[n]) != EOF; n++); /\*EOF: End Of File\*/

fclose(inputfile);

//calculate mean data

mean = fmean(xdata, n);

//calculate variance

var = fvariance(xdata, n);

//calculate skew of n

skewN = fskewn(xdata, n);

//prompt user for output file name

printf("\nOutput file name: ");

scanf("%s", &outputName);

//write to the file

outputfile = fopen(outputName, "w");

fprintf(outputfile, "mean:\t\t%3.3g\n", mean);

fprintf(outputfile, "variance:\t%0.3g\n", var);

fprintf(outputfile, "skew:\t\t%0.3g\n", skewN);

system("PAUSE");

//use MATLAB to create histogram

Engine \*ep;

if ((ep = engOpen(NULL)) == NULL) return 1;

mxArray\* matx1 = mxCreateNumericMatrix(n, 1, mxSINGLE\_CLASS, mxREAL);

memcpy((void \*)mxGetPr(matx1), (void \*)xdata, sizeof(float) \* 100);

engPutVariable(ep, "xdata", matx1);

engEvalString(ep, "fig = figure; histogram(xdata)"

"xlabel('Bins'); ylabel('Frequency');"

"print(fig,'histogram','-dpng')");

system("PAUSE");

engClose(ep);

//system("PAUSE");

return 0;

}

/\* lab2part3.cpp : Defines the entry point for the console application.

Peter Dobbs

BIEN 3200 - Section 401

13 September 2016

Lab 2

Program Description:

This program reads clinical sinus rhythm (normal) ECG data from

a text file (sinus2.txt) and uses a connection with MATLAB to plot

the data on a graph

\*/

#include "stdafx.h"

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include <string.h>

#include "engine.h"

#include "mex.h"

int main(void) {

printf("Lab 2, Part 3");

int length;

static float xdata[2500], ydata[2500];

FILE \*fp;

//read from file

fp = fopen("sinus2.txt", "r");

if (fp == NULL) { //file not found

printf("Input file not found!\n");

system("PAUSE");

return 1; //end program

}

for (length = 0; fscanf(fp, "%g %g", &xdata[length], &ydata[length]) != EOF; length++); /\*EOF: End Of File\*/

fclose(fp);

//interface with Matlab for plotting

Engine \*ep;

ep = engOpen(NULL);

mxArray\* matx1 = mxCreateNumericMatrix(length, 1, mxSINGLE\_CLASS, mxREAL);

mxArray\* maty1 = mxCreateNumericMatrix(length, 1, mxSINGLE\_CLASS, mxREAL);

/\* Copies values of x1 and y1 into the allocated space: copies 10 \*(# of bytes for a float) bytes

from x1 to matx1 or from y1 to maty1 \*/

memcpy((void \*)mxGetPr(matx1), (void \*)xdata, sizeof(float) \* length);

memcpy((void \*)mxGetPr(maty1), (void \*)ydata, sizeof(float) \* length);

engPutVariable(ep, "X", matx1);

engPutVariable(ep, "Y", maty1);

engEvalString(ep, "figure; plot(X,Y);"

"title ('Amplitude vs Time');"

"xlabel ('Time (sec)');"

"ylabel ('Amplitude (mV)');");

system("pause");

engClose(ep);

return 0;

}

**Flow Charts:**

